DIFFERENTIALLY PRIVATE BYTE PAIR ENCODING TOKENIZERS FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tokenizers are a standard part of large language model (LLM) releases, providing practical tools for splitting text and counting tokens. However, vocabulary and ordered merge list released with common byte pair encoding (BPE) tokenizers for LLMs reflect recurring patterns in the data corpus used during their training. We ask whether access to such a tokenizer can reveal that a particular website was included in its training data. Under three access models, exposing (1) token counts only, (2) token identities, and (3) the full BPE tokenizer with both the vocabulary and the ordered merge list, we design website-level membership inference attacks calibrated using shadow tokenizers trained with and without the target website. Under the most restrictive token-count access, our attack reaches 60.9% TPR at 1% FPR, while the best prior attack under the same access stays at 1.3% and therefore does no better than guessing. Motivated by this leakage, we also evaluate defenses. We find that simple low-count filtering reduces the attack success under token-count access but does not reliably protect against the stronger token-identity and merge-list attacks. Moreover, prior defenses provide no provable guarantees on privacy leakage. We therefore propose DP-BPE, the first differentially private BPE training algorithm, which gives the released vocabulary and merge list a provable website-level privacy guarantee. At a 4% relative utility cost, it reduces every attack we evaluate to close to random guessing, while prior defenses leak substantially more. We also open-source our DP-BPE as a Rust implementation for the community. Together, our findings highlight the need for privacy-aware tokenizer training and show that a website-level privacy guarantee is possible at a small utility cost.

1 INTRODUCTION

Large language models (LLMs) rely on tokenizers that convert input text into sequences of discrete tokens. Since the model is trained on this representation, using it correctly requires the same text-to-token mapping at inference time. Tokenizers are therefore commonly released with models to support reproducible encoding, API-cost estimation, prompt-length accounting, and local inference. Yet tokenizers are not merely static preprocessing code. Instead, they are usually trained on large text corpora, often drawn from the same web-scale sources used to train the corresponding LLM (Tong et al., 2026). A released tokenizer can therefore carry information about the data used to train the model even when the model weights, gradients, and corpus are not disclosed. Byte pair encoding (BPE) (Sennrich et al., 2016) is the standard tokenizer algorithm for modern LLMs, used across open-weight and proprietary model families (Grattafiori et al., 2024; Olmo et al., 2025; Team et al., 2025; Qwen Team, 2026a). BPE repeatedly merges frequent adjacent symbols in the training corpus. The result is a set of tokens together with an ordered merge list recording which patterns were selected earlier or later during training. Both the vocabulary and the merge list therefore reflect the statistics of the training corpus, such as common words, formatting conventions, and domain-specific identifiers.

We study whether this information leaks *website-level membership*, *i.e.,* whether access to a tokenizer reveals that a particular website was used to train it. This is membership in the tokenizer's training corpus, which generally overlaps with the model's training corpus (Black et al., 2022; Biderman et al., 2023). Even when website text is public, its inclusion in a particular training corpus is usually not. Such leakage can expose data-sourcing decisions, licensing or opt-out compliance, and internal provenance information. Our question is: *given access to a tokenizer and a candidate website,*

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.

can an adversary infer whether the website was included in tokenizer training? Prior work shows that tokenizers reveal aggregate mixture properties (Hayase et al., 2024) and support membership inference attacks (MIAs) to some extent (Tong et al., 2026).

In this paper, we first show that the actual risk is larger than observed by these prior works. Therefore, we systematically exploit signals exposed by BPE training. In particular, we consider the signals based on *three* access levels that an adversary can obtain in practice. Under *Token-Count Access*, the adversary observes only the number of tokens produced for chosen text. This corresponds to the setting of a commercial API that reports token usage without releasing the tokenizer, as Claude (Anthropic, 2026) and Gemini (Google, 2026) do through their token-counting API. Under *Token-Identity Access*, the adversary additionally observes the vocabulary.[1] Under *Merge-Order Access*, the adversary observes the full merge list, whose order reflects the priority the tokenizer learned during training, which is what, for example, open-weight model releases provide. We exploit these signals to design calibrated MIAs against tokenizer training data, using shadow tokenizers trained with and without the candidate website. As Figure 1 shows, evaluations based only on prior MIAs (Prior) substantially understate tokenizer leakage over all setups. For example, at a vocabulary of 200,000, our strongest MIA reaches 60.9% TPR@1%FPR, against 17.8% for the strongest prior MIA under any of the three access models.

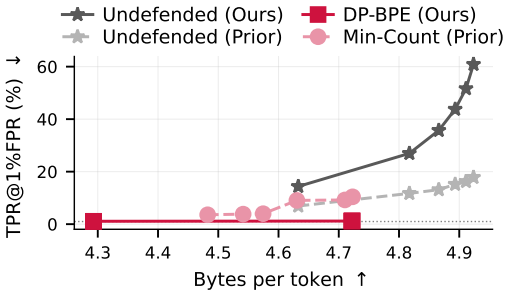

Figure 1: **Prior MIAs underestimate *Undefended* tokenizer leakage, and our DP-BPE removes it at a significantly smaller utility cost than prior defenses (Min-Count).** We report leakage (as TPR@1%FPR) against the tokenizer utility (in Bytes per token).

Motivated by this leakage, we also study defenses. Prior work proposes heuristic defenses (Tong et al., 2026). However, there are no defenses that provide provable protection guarantees. To close this gap, we propose DP-BPE, a differentially private (Dwork et al., 2006) BPE training algorithm with a provable bound on the leakage. As highlighted in Figure 1, DP-BPE substantially reduces leakage across all three access models while preserving high utility.

In summary, we make the following contributions:

1. We decompose the tokenizer attack surface into three access models that follow from the structure of a BPE release, namely the merge order, the vocabulary, and the token counts an API must expose, and show that each one supports website-level MIAs on its own.
2. We propose MIAs for each access model and show that they reveal substantially more leakage than reported in prior work, and that the heuristic defenses proposed against prior MIAs do not withstand our stronger attacks.
3. We show that leakage does not require releasing the tokenizer, since an adversary observing only the token counts returned by a billing endpoint already infers membership.
4. We propose DP-BPE, a differentially private BPE training algorithm and the first to give the released vocabulary and merge list a provable website-level privacy guarantee at a small cost in bytes per token.
5. We build DP-BPE trainer as a Rust extension of the `tokenizers` library and will open-source it for the community upon acceptance. The code is in the supplementary material for review.

2 Background and Related Work

Tokenizers. Tokenizers map raw text into sequences of symbols used by LLMs. Several subword families exist, such as WordPiece (Wu et al., 2016) and the unigram language model of SentencePiece (Kudo, 2018; Kudo & Richardson, 2018), but virtually all modern LLMs tokenize with BPE (Grattafiori et al., 2024; Olmo et al., 2025; Team et al., 2025; Qwen Team, 2026a;b; Olmo et al.,

[1] That a vocabulary carries traces of its training corpus is already visible in released tokenizers. The OpenAI vocabulary contains tokens such as `SolidGoldMagikarp` and `davidjl`, the usernames of Reddit users whose posts were frequent enough in the training data to earn a dedicated token (Rumbelow & Watkins, 2023).

2025). BPE starts from a base alphabet, often characters or bytes, and iteratively adds merge rules for frequent adjacent symbol pairs (Sennrich et al., 2016). Each merge is chosen to shorten the encoding of the training corpus, so BPE training is driven by compression, and the resulting tokenizer is fitted to the text it was trained on. Appendix B reviews the main subword tokenizer families and gives a reference BPE training loop. After training, a BPE tokenizer consists of a vocabulary and an ordered merge list that records which pair patterns were prioritized during training.

Membership inference attacks (MIAs). In machine learning, MIAs ask whether a particular data sample was included in the training data of a machine learning model (Shokri et al., 2017). Early MIAs (Shokri et al., 2017; Watson et al., 2022; Ye et al., 2022) predict membership in a target model by training multiple shadow models, *i.e.,* models trained on datasets with known member and non-member data. By comparing the shadow models' behavior on their training members versus on the non-members, one can train an attack classifier that, given the target model and a candidate data sample, predicts whether this sample was a member. Later MIAs, such as LiRA (Carlini et al., 2022), refine this approach by modeling the output distributions of the shadow models and performing a likelihood-ratio test. The unit of membership does not necessarily need to be a single sample. For example, document-level MIAs target an entire book or article (Meeus et al., 2024), and dataset inference a whole collection of those (Maini et al., 2021; Dziedzic et al., 2022; Maini et al., 2024). Focusing on BPE tokenizers, Tong et al. (2026) study website-level MIAs based on shadow tokenizers. Their attacks exploit the compression rate a tokenizer achieves on the candidate website, the overlap between the website's frequent strings and the vocabulary, and the position of website-relevant tokens in the merge list. We build on this approach, but design stronger MIAs that reveal substantially more leakage by calibrating the signals effectively.

Differential privacy (DP) (Dwork et al., 2006; Dwork & Roth, 2014) is the standard formal framework for bounding the privacy leakage by ensuring that the inclusion or removal of a single element, e.g., a sample or, as in our case, an entire website does not noticeably change the output of a randomized algorithm. Formally, a randomized algorithm $\mathcal{M}$ satisfies (ε, δ)-DP if for all datasets $\mathcal{D}, \mathcal{D}'$ differing in at most one element, and all outputs S,

$$\Pr[\mathcal{M}(\mathcal{D}) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta, \tag{1}$$

where ε is the privacy budget and δ is a small probability of failure. DP is preserved under *post-processing*, so anything computed from a private release stays private without consuming further budget. Our website-level DP tokenizer builds on results from prior work on DP histogram release, namely the Gaussian Sparse Histogram Mechanism (Karrer et al., 2024), which hides both the counts (Gopi et al., 2020; Kim et al., 2021) and elements of a released histogram.

Tokenizer defenses. Prior DP tokenizers, built for WordPiece (Hoory et al., 2021) and SentencePiece (Ponomareva et al., 2022), protect individual *sentences* rather than *entire websites*. Given that information is usually spread across an entire website and not only in a single sentence, providing website level privacy is more relevant, but also a more challenging setup. While defenses for BPE at the website level exist (Tong et al., 2026), they rely purely on heuristics, such as filtering infrequent tokens or randomizing the merge order. Therefore, they do not provide formal privacy guarantees, and we show that they do not withstand our stronger MIAs. In contrast, we propose DP-BPE, a DP BPE training algorithm that provides a formal privacy guarantee at the website level and provides strong protection while maintaining high utility.

3 Access Models and Problem Definition

Website-level MIAs. We study website-level MIAs against trained tokenizers. Let $\mathcal{W}$ denote a universe of websites, let $\mathcal{S} \subseteq \mathcal{W}$ denote the unknown subset used for tokenizer training, and let $w^\star \in \mathcal{W}$ be a candidate website with associated text collection $D_{w^\star}$. The adversary tests

$$H_1 : w^\star \in \mathcal{S} \qquad \text{versus} \qquad H_0 : w^\star \notin \mathcal{S}. \tag{2}$$

Let $T^\star$ denote the target tokenizer trained on the unknown subset $\mathcal{S}$. The adversary does not necessarily observe the full tokenizer. We write $\mathcal{O}(T^\star)$ for the part of the target tokenizer visible under the adversary's access model. An attack is a procedure $\mathcal{A}$ that takes the observable tokenizer information together with the candidate website text and returns a real-valued score

$$s = \mathcal{A}(\mathcal{O}(T^\star), D_{w^\star}).$$

A threshold on s yields the final membership decision. We focus on website-level membership because tokenizer training is driven by aggregate frequencies, because web-scale pipelines make crawling, licensing, and opt-out decisions such as `robots.txt` per website.

Adversary access models. A trained BPE tokenizer is a *vocabulary* together with the *order* in which its tokens were merged, so a deployment can release neither, one, or both. Releasing neither corresponds to the *Token-Count* access. The release of the vocabulary only is denoted as the *Token-Identity* access. The full tokenizer release is represented as the *Merge-Order* access. Next, we give more details about each access level and provide a summary in Table 4 in Appendix C.

Token-Count access. The adversary can submit chosen text and observe only the *number* of produced tokens, but no vocabulary or token strings. This is realistic for hosted model APIs, which expose token counts through billing, context-window accounting, or usage metadata. The leaked signal relies on compression: if a website influenced tokenizer training, its characteristic strings may be represented more compactly. We write

$$\mathcal{O}_{\mathrm{count}}(T^\star) : \mathcal{X} \to \mathbb{N},$$

where $\mathcal{O}_{\mathrm{count}}(T^\star)(x)$ is the number of tokens produced when encoding x with the tokenizer $T^\star$.

Token-Identity access. The adversary can encode chosen text and observe the produced token identities, such as token strings, and may also know the vocabulary as an unordered set, but not the merge order. This is realistic for client-side tokenizer libraries and APIs that return tokenized text for debugging. It reveals which website-specific strings entered the vocabulary. We write

$$\mathcal{O}_{\mathrm{id}}(T^\star) : \mathcal{X} \to V(T^\star)^*,$$

where $V(T^\star)$ is the released vocabulary and $\mathcal{O}_{\mathrm{id}}(T^\star)(x)$ is the sequence of token identities emitted for input x.

Merge-Order access. This is the standard BPE release, realistic for open models, where tokenizer files are distributed for reproducibility and interoperability. It is a strict superset of Token-Identity access, where the adversary can exploit both whether a token exists and where it appears relative to other candidate tokens. We write

$$\mathcal{O}_{\mathrm{order}}(T^\star) = \Pi(T^\star),$$

where $\Pi(T^\star)$ denotes the released merge order, or equivalently the released vocabulary written in the order the tokenizer learned during training.

In all three access models, we consider the standard MIA assumptions (Shokri et al., 2017; Carlini et al., 2022; Zarifzadeh et al., 2024), adopted for tokenizers by Tong et al. (2026). In particular, the adversary knows the candidate website text $D_{w^\star}$ and can have partial knowledge of the tokenizer-training procedure, but knows neither the other websites in the target's training set nor its training seed. The adversary also samples auxiliary websites from the same distribution as the target's training data and trains shadow tokenizers that include the candidate website in some runs and exclude it in others, yielding IN and OUT references for calibration. Auxiliary websites are obtainable because tokenizers are trained on web text, which the adversary can crawl. Shadow tokenizers are cheap to train, since one run takes minutes rather than weeks of compute required to train its language model.[2] This makes the tokenizer an easier target than the model trained on the same corpus.

4 OUR WEBSITE-LEVEL TOKENIZER MIAS

We now instantiate our new MIAs for the three access models outlined in the previous section. The attacks differ in which tokenizer signal they observe, namely token counts, token identities, or merge order by comparing the target tokenizer against shadow tokenizers trained with and without the candidate website. The adversary trains shadow tokenizers $\{T_i\}_{i=1}^N$ once, each on a random half of the website universe $\mathcal{W}$, and for a candidate website w defines

$$\mathcal{I}_w^{\mathrm{in}} = \{i : w \text{ is included in shadow run } i\},$$
$$\mathcal{I}_w^{\mathrm{out}} = \{i : w \text{ is excluded in shadow run } i\}.$$

[2]Training one BPE tokenizer with a vocabulary of 200,000 tokens on 10,545 websites takes around 3 minutes on a 2×56-core Xeon Gold 5420+ CPU, so the 128 shadow tokenizers used throughout our evaluation cost under 7 hours of serial compute.

Deciding membership then amounts to a hypothesis test between the IN and the OUT distribution of a statistic extracted from the tokenizer. We use the Neyman-Pearson lemma (Neyman & Pearson, 1933) to motivate our MIAs, as LiRA (Carlini et al., 2022) does for MIAs on trained models. The lemma states that the most powerful test at any fixed false positive rate is the likelihood ratio test. Our MIAs therefore score a candidate website with the log-likelihood ratio

$$s(w) = \log p\Big(x^* \mid \widehat{\theta}_w^{\text{in}}\Big) - \log p\Big(x^* \mid \widehat{\theta}_w^{\text{out}}\Big), \tag{3}$$

where $x^\star = x(T^\star, D_w)$ is the statistic that extracts from the target tokenizer $T^\star$ on the website text D_w, and $\widehat{\theta}_w^{\text{in}}$ and $\widehat{\theta}_w^{\text{out}}$ are reference models fitted on the statistics of the shadow tokenizers indexed by $\mathcal{I}_w^{\text{in}}$ and $\mathcal{I}_w^{\text{out}}$. The following MIAs instantiate Equation (3) for each access model. Appendix C provides the full algorithms and estimators.

4.1 TOKEN-COUNT ATTACK

Under *Token-Count* access, the adversary observes only how many tokens the tokenizer emits for chosen text. This is the weakest signal we consider, but it is also the one most directly aligned with the tokenizer's own training objective: BPE greedily selects the merges that most reduce the number of tokens needed to represent its training corpus, so compression is exactly the quantity the tokenizer was optimized for on the data it saw. The statistic is therefore the scalar compression signal, measured in *bytes per token* (BPT),

$$\text{BPT}(T, D_w) = \frac{\text{Bytes}(D_w)}{\text{TokenCount}(T(D_w))}. \tag{4}$$

A higher BPT means that the website text is represented more compactly. If w influenced tokenizer training, its characteristic strings may receive more efficient segmentations, shifting the IN and OUT compression distributions. We fit a Gaussian distribution over the BPT scores of the IN shadows and another over those of the OUT shadows, and apply Equation (3).

4.2 TOKEN-IDENTITY ATTACK

Under *Token-Identity* access, the adversary observes the vocabulary $V(T)$. Vocabulary slots are limited and go to the most frequent strings in the training corpus, so a string that only w uses enters $V(T)$ only if w was part of that corpus. Let U_w be the set of tokens the target tokenizer $T^\star$ emits when encoding D_w. For each token $t \in U_w$, the observed statistic is its presence $\mathbf{1}[t \in V(T)]$ in the vocabulary. We model the presence of each token as an independent Bernoulli variable, whose IN and OUT probabilities q_t^{in} and q_t^{out} we estimate as the fraction of IN and OUT shadow tokenizers containing t. Following the same likelihood-ratio motivation as Equation (3), this yields the score

$$s_{\text{id}}(w) = \sum_{t \in U_w} \left[\log q_t^{\text{in}} - \log q_t^{\text{out}}\right]. \tag{5}$$

In practice, we retain only tokens whose IN and OUT presence probabilities are well separated. Appendix C.2 gives the derivation with the candidate construction and filtering.

4.3 MERGE-ORDER ATTACK

Under *Merge-Order* access, the adversary observes the merge order in addition to the vocabulary. BPE merges the most frequent pair at each step, so a token's rank correlates with how often its string occurred in the training corpus. Presence of a token records only that its frequency exceeded a threshold, whereas rank records approximately what that frequency was. Ordering evidence has no analogue in MIA against trained models, which expose losses and confidences but no ranking over their training statistics, so we construct its reference model below. Let $r(t)$ be the rank of token t in the merge order of the target tokenizer $T^\star$, with smaller rank meaning merged earlier. The attack signal is the relative order of pairs (a, b) of tokens in U_w, collected in a pair set P_w. Our first variant, *Bernoulli*, models each pair comparison as an independent Bernoulli variable and estimates its probabilities q_{ab}^{in} and q_{ab}^{out} as the fraction of IN and OUT shadow tokenizers in which a precedes b. Independence is an approximation here, since ranks form one total order and the comparisons are

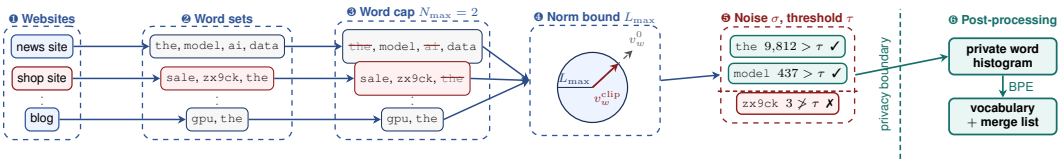

Figure 2: **DP-BPE privatizes the word histogram, then trains BPE on it.** ❶ Each website is ❷ reduced to the *set* of words it uses, then bounded by ❸ a cap of $N_{\max}$ words per site, drawn uniformly at random and set to 2 here, and ❹ a rescaling onto the ball of radius $L_{\max}$. ❺ DP-BPE sums the rescaled contributions of the websites using each word, adds Gaussian noise of scale σ, and keeps the words whose noisy sum exceeds τ. ❻ Only this histogram depends on the corpus, so BPE training is post-processing and the released tokenizer inherits the guarantee at any vocabulary size.

therefore transitively constrained. Writing $y_{ab} = \mathbf{1}[r(a) < r(b)]$ for the order bit observed on the target tokenizer, we have

$$s_{\text{order}}(w) = \sum_{(a,b)\in P_w} \log \frac{(q_{ab}^{\text{in}})^{y_{ab}}(1-q_{ab}^{\text{in}})^{1-y_{ab}}}{(q_{ab}^{\text{out}})^{y_{ab}}(1-q_{ab}^{\text{out}})^{1-y_{ab}}}. \tag{6}$$

We also evaluate a Bradley-Terry variant (Bradley & Terry, 1952), which uses this structure by replacing the independent pair probabilities with a shared latent ranking model. Appendix C.3 gives the derivations for both variants together with the pair construction and filtering.

5 Our Differentially Private Tokenizers

Since no prior defense gives BPE tokenizers a formal privacy guarantee at the website level, we propose DP-BPE, a DP training algorithm for BPE that provides this guarantee for the trained tokenizer. Figure 2 illustrates the six steps ❶–❻ of DP-BPE, marked below. Our approach concentrates the entire privacy cost into a single release of a private histogram of words and then trains a BPE tokenizer on that histogram as a post-processing step. The guarantee is therefore independent of the target vocabulary size and the number of merges performed. Appendix D gives the full algorithms and calibration details.

Per-website clipping. We represent each website w ❶ by the *set* of words it contains ❷, i.e., by the binary vector $v_{w,j} = \mathbf{1}[\text{word } j \text{ occurs in website } w]$, where the vector v_w per each website w is of the size of all possible words. The first step is to bound the influence of any one website on the aggregate. We therefore ℓ_0 cap each website at $N_{\max}$ words ❸, keeping a uniformly random subset when a site exceeds the budget, and then rescale the resulting vector to Euclidean norm (ℓ_2) at most $L_{\max}$ ❹. We denote the resulting clipped contribution by v_w^{clip}. The subset is drawn from that website's own words alone, so it does not affect the noise scale or the threshold. The word cap bounds how many new words a single website can introduce, which is what the calibration of the release threshold rests on. The norm bound limits the *sensitivity*, i.e., how much each website can influence the final outcome to $L_{\max}$. Limiting sensitivity is crucial to achieve DP and allows to correctly calibrate the noise addition. We discuss these parameters more in Appendix D.1.

Private release. The previous step produces a clipped contribution v_w^{clip} for each website w. We aggregate these contributions into the histogram $a = \sum_{w\in\mathcal{D}} v_w^{\text{clip}}$ and release it in a differentially private manner ❺. We rely on the Gaussian Sparse Histogram Mechanism (Karrer et al., 2024), which is designed for high-dimensional histograms with a small number of non-zero entries. For each observed word j, we add independent Gaussian noise ξ_j and release the word only if its noisy count exceeds a threshold τ:

$$\tilde{a}_j \;=\; a_j + \xi_j, \quad \xi_j \sim \mathcal{N}(0,\sigma^2), \quad \mathcal{S} \;=\; \{\, j : \tilde{a}_j > \tau \,\}.$$

This mechanism guards against two distinct kinds of leakage. The noise protects the *counts* of words present in both neighboring corpora, with σ calibrated to the norm bound $L_{\max}$. The threshold protects *which words are released*, since a word contributed by one website alone would otherwise be revealed by its mere appearance in the output, and we calibrate it so that a union bound over the at most $N_{\max}$ words a single website can introduce keeps this event below δ_{sel}. The privacy budget is split as $\delta = \delta_{\text{sel}} + \delta_{\text{cnt}}$ across the two modes. Appendix D.2 gives the calibration.

Table 1: **Our MIAs outperform prior work under every access model and at every vocabulary size.** TPR@1%FPR against undefended tokenizers. Bold marks the strongest MIA in each column within its access model. Table 5 reports ROC-AUC for the same MIAs and vocabulary sizes.

Access model	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
		40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	Compression Rate (Tong et al., 2026)	1.2±0.1	1.2±0.0	1.2±0.0	1.2±0.0	1.2±0.0	1.3±0.0
	Token-Count Attack (ours)	**6.4±0.1**	**18.6±0.1**	**29.2±0.2**	**41.1±0.3**	**51.6±0.3**	**60.9±0.2**
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	6.9±0.1	11.7±0.2	13.1±0.3	15.2±0.3	16.3±0.3	17.8±0.3
	Token-Identity Attack (ours)	**9.6±0.1**	**19.0±0.1**	**27.0±0.1**	**35.7±0.1**	**44.3±0.2**	**52.3±0.2**
Merge-Order	Frequency (Tong et al., 2026)	2.3±0.1	6.0±0.1	8.9±0.1	12.3±0.1	13.4±0.2	10.9±0.4
	Naive Bayes (Tong et al., 2026)	3.1±0.4	2.4±0.0	3.6±0.1	6.4±0.3	7.2±0.5	8.5±0.4
	Merge-Order with Bernoulli (ours)	**14.4±0.1**	**27.0±0.2**	**35.7±0.2**	**43.8±0.3**	**50.7±0.3**	**56.2±0.2**
	Merge-Order with Bradley-Terry (ours)	11.6±0.1	21.8±0.2	29.0±0.3	37.2±0.3	43.9±0.3	50.2±0.2

Privacy analysis. We now show that the released tokenizer is differentially private with respect to the addition or removal of one entire website. The guarantee is on the released tokenizer itself, so it covers all three access models at once. Theorem 1 states the resulting guarantee informally.

Theorem 1 (Website-level privacy of DP-BPE, informal)**.** *For any $\varepsilon > 0$ and $\delta_{\text{sel}}, \delta_{\text{cnt}} > 0$, there exist a noise scale σ and threshold τ such that the tokenizer released by DP-BPE is $(\varepsilon, \delta_{\text{sel}} + \delta_{\text{cnt}})$-DP at the website level.*

The proof uses three ingredients, in the order the algorithm applies them. The two clips of ❸–❹ bound how much one website can change the aggregate, the Gaussian Sparse Histogram Mechanism (Karrer et al., 2024) gives the guarantee for the release of ❺, and closure of DP under post-processing covers the BPE training of ❻. Theorem 2 in Appendix D.2 gives the full formal statement, including how σ and τ are calibrated, and the proof.

6 Empirical Evaluation

6.1 Setup

We attack BPE tokenizers trained on 21,090 websites from the C4 corpus (Raffel et al., 2020), with 10 target and 128 shadow tokenizers per setting, each trained on a uniformly random half of the websites, at six target vocabulary sizes from 40,000 to 200,000. Our three MIAs and the four of Tong et al. (2026) attack both undefended tokenizers and tokenizers defended by the Min-Count filtering and Softmax BPE of Tong et al. (2026) or by our DP-BPE. Following Carlini et al. (2022) we report the true positive rate at a 1% false positive rate (TPR@1%FPR), write ± for the standard error over all targets, and measure tokenizer utility as BPT on the websites each target was not trained on. Appendix E describes every baseline, Appendix F the full protocol and every parameter, and Appendix G the receiver operating characteristic (ROC) curves and the area under them (ROC-AUC).

6.2 Attack Results

Main results. Our MIAs outperform the prior tokenizer MIAs under all three access models, and they do so at every vocabulary size we evaluate. Table 1 reports TPR@1%FPR against undefended tokenizers. At the largest vocabulary, 200,000, our Token-Count Attack reaches 60.9% where the strongest attack from prior work, Vocabulary Overlap, reaches 17.8%. Over all setups, our MIAs yield at least a 1.4× improvement over prior MIAs. Which of our MIAs leads depends on how large the vocabulary is. Merge-Order with Bernoulli is the strongest up to 140,000, rising from 14.4% to 43.8% over that range, and the Token-Count Attack overtakes it above. We hypothesize that Token-Count overall yields the highest success rate at large vocabularies because it captures the exact signals that tokenizers are trained for, namely compressing the token representations.

Additionally, we observe that leakage grows with vocabulary size under every access model. Between 40,000 and 200,000 the Token-Count Attack rises from 6.4% to 60.9% and Merge-Order with Bernoulli from 14.4% to 56.2%, so the interface that reveals the least and the one that reveals the most both identify more than half the member websites at a 1% false positive rate. The prior MIAs grow far more slowly over the same range, with Vocabulary Overlap moving only from 6.9% to

Table 2: **DP-BPE gives better privacy-utility tradeoffs.** TPR@1%FPR for our MIAs at a vocabulary of 200,000, with BPT in the last row and bold marking the strongest MIA in each column. Appendix I.3 reports every MIA, with Min-Count and DP-BPE at all six vocabulary sizes.

Attack	Undefended	Min-Count (Tong et al., 2026)			Softmax BPE (Tong et al., 2026)			DP-BPE (ours)		
		$n_{\min}$=32	$n_{\min}$=64	$n_{\min}$=256	$\lambda = 1$	$\lambda = 10$	$\lambda = 20$	$\varepsilon = 1$	$\varepsilon = 3$	$\varepsilon = 10$
Token-Count Attack	**60.9±0.2**	10.9±0.2	6.0±0.2	3.0±0.1	1.0±0.1	1.0±0.1	**58.6±0.3**	**1.1±0.1**	1.0±0.0	**1.2±0.1**
Token-Identity Attack	52.3±0.2	31.3±0.1	18.1±0.2	6.4±0.1	**4.6±0.1**	**4.6±0.1**	46.4±0.2	0.9±0.0	1.1±0.1	1.1±0.0
Merge-Order with Bernoulli	56.2±0.2	**38.8±0.3**	**26.2±0.3**	**11.4±0.2**	4.3±0.1	4.5±0.1	50.4±0.2	0.8±0.0	0.7±0.0	0.4±0.0
Merge-Order with Bradley-Terry	50.2±0.2	33.8±0.3	22.1±0.3	9.0±0.1	3.5±0.0	3.6±0.0	45.5±0.2	1.0±0.1	**1.1±0.0**	1.0±0.0
BPT	4.92±0.01	4.66±0.01	4.42±0.01	3.76±0.01	1.22±0.01	3.51±0.48	4.92±0.01	3.73±0.01	4.29±0.01	4.72±0.01

17.8%, so a larger vocabulary does not make membership easier to infer for every adversary, only for one that calibrates per website, like our MIAs. Current LLM tokenizers have up to 256,000 tokens (Appendix B), so the vocabularies released today are among those that leak the most in our experiments. Below 40,000 the leakage keeps falling, to 1.1% for the Token-Count Attack and 4.8% for Merge-Order with Bernoulli at a vocabulary of 10,000 (Appendix G.2). For completeness, in Appendix G.1, we report the whole false positive range rather than at 1% alone.

Document-level membership. We also evaluate whether our MIAs detect membership when each website contributes a single page, meaning that website membership coincides with membership of that document. In this setting, the Token-Identity Attack reaches 84.1% TPR@1%FPR (Appendix H).

Why token counts alone leak. The gap between Compression Rate and our Token-Count Attack in Table 1 shows that raw compression alone is a weak membership score. We hypothesize that websites differ in how compactly they tokenize, since some contain short common words and simple formatting while others contain identifiers, unusual strings, or boilerplate, so a threshold shared across websites confounds membership with properties of the text. Figure 3 illustrates this at a vocabulary of 80,000. For some websites the IN and OUT shadow distributions almost overlap, so token counts carry little evidence, while for others they separate clearly, making membership visible without observing a single token identity. Our MIA compares each website against its own shadow tokenizers, which is why it reaches 18.6% TPR@1%FPR while Compression Rate, which uses a similar signal without this calibration, stays close to random guessing.

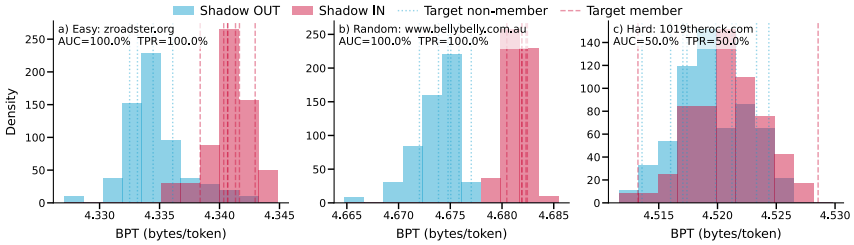

Figure 3: **How much token counts reveal depends on the website.** Token-Count score distributions at a vocabulary of 80,000 for the websites of highest (Easy) and lowest (Hard) per-website ROC-AUC and one drawn uniformly (Random).

6.3 DEFENSE RESULTS

We now measure how much of this leakage each defense removes and what it costs in utility.

Prior heuristic defenses. Table 2 shows that both prior heuristic defenses have to substantially decrease the utility of the tokenizer before leakage falls meaningfully. Min-Count reduces leakage monotonically in $n_{\min}$, but its largest threshold, $n_{\min} = 256$, drops BPT from 4.92 to 3.76, and Merge-Order with Bernoulli still identifies 11.4% of the member websites there. Softmax BPE at $\lambda = 1$ limits all of our MIAs to at most 4.6%, but its BPT collapses to 1.22, and at $\lambda = 20$, where utility returns to the undefended level, so does the leakage. Min-Count's protection is also uneven, since $n_{\min} = 32$ reduces the Token-Count Attack to 10.9% but leaves Merge-Order with Bernoulli at 38.8%, more than twice what the strongest prior MIA, Frequency, measures (Appendix I.3).

Our DP-BPE. DP-BPE exhibits low privacy leakage across all privacy budgets. At $\varepsilon = 10$, Merge-Order with Bernoulli drops from 56.2% TPR@1%FPR undefended to 0.4% and the Token-Count Attack from 60.9% to 1.2%, so no MIA does better than guessing at this false positive rate. However, tightening the privacy budget to $\varepsilon = 1$ decreases the utility to 3.73 BPT from 4.72 at $\varepsilon = 10$. Between these budgets BPT changes smoothly, with 4.29 at $\varepsilon = 3$, so even at $\varepsilon = 1$ DP-BPE retains most of the utility, 76% of the 4.92 BPT of the undefended tokenizer (Appendix I.2). Overall, the results in Table 2 clearly indicate that our DP-BPE defense exhibits the best privacy-utility trade-offs.

6.4 Ablation Study

We now ablate five assumptions, a matching vocabulary size, training rule and number of pages per website, 128 shadow tokenizers, and shadow tokenizers trained with the candidate website.

Mismatched vocabulary size. An adversary rarely knows the vocabulary size of the target, so its shadow tokenizers are built at some other size. Fixing the shadows at $110{,}000$ and sweeping the target vocabulary, performance peaks at the matched size and degrades in both directions without falling to chance, since a shadow tokenizer that stops its merge list at a different point spans a learns a vocabulary that only partially overlaps the target's. Merge-Order with Bernoulli keeps 10.3% TPR@1%FPR against a target at $200{,}000$, down from 35.7% when the two match (Appendix G.3). The adversary can avoid this mismatch by estimating the vocabulary size from token counts alone, querying the tokens of its own shadow tokenizers in the order they were learned and locating where the target stops encoding them as single tokens (Algorithm 8). With at most 544 queries, the estimate is off by 1.08% on average at a vocabulary of $110{,}000$ (Appendix G.4).

Mismatched defense. The adversary may also not know whether the target was defended at all. Keeping the Min-Count-defended target but training the shadow tokenizers without the defense, the Token-Count Attack collapses to the 1% FPR at every threshold, so what protection Min-Count offers under Token-Count access rests on the adversary not knowing the threshold. Merge-Order with Bernoulli is unaffected, reaching 31.6% TPR@1%FPR without the matching shadows against 31.9% with them at $n_{\min} = 32$ and a vocabulary of $110{,}000$, so Min-Count does not protect the merge list even when $n_{\min}$ is public (Appendix I.1).

Mismatched pages per website. The adversary also does not know how many pages of each website the target was trained on, and at a vocabulary of $200{,}000$, each of our MIAs is strongest when the shadow tokenizers are trained on the same number. With matching shadow tokenizers, fewer pages per website make each of our MIAs stronger, so they do not rely on a website contributing much text. Overestimating that number costs less than underestimating it (Appendix G.5).

Reduced numbers of shadow tokenizers. Frequency and Naive Bayes saturate by eight shadow tokenizers, whereas each of our MIAs peaks at 128, with the Token-Identity Attack rising from 1.4% TPR@1%FPR at four shadow tokenizers to 27.0% at a vocabulary of $110{,}000$ (Appendix G.6).

Offline setting. Tokenizers remain vulnerable when the adversary uses only shadow tokenizers trained without the candidate website, the offline setting (Carlini et al., 2022), in which each MIA tests how far the target's statistic lies from the OUT reference model alone. The Token-Count Attack keeps at least 95% of its TPR@1%FPR from Table 1 at every vocabulary size, reaching 58.0% at $200{,}000$, where the Token-Identity Attack reaches 44.1% (Appendix G.7).

7 Conclusions

In this work, we formulate calibrated website-level MIAs against BPE tokenizers under three access models and identify significant leakage under all of them. Token-Count Access is particularly practical because a provider can withhold its tokenizer while still exposing token counts through an API; under this access model, our MIA succeeds where the prior MIA does no better than guessing. This signal becomes useful after calibrating each website against shadow tokenizers trained with and without it. To mitigate leakage, we introduce DP-BPE, a differentially private BPE training algorithm and the first to provide a provable upper bound on website-level leakage. We empirically verify that DP-BPE reduces the performance of all MIAs to random chance, while maintaining high utility for the privately released tokenizers. By providing a practical open-source implementation of DP-BPE in Rust for the community, our work contributes toward privacy-preserving tokenizer release.

AI USE STATEMENT

In this work, we used generative AI tools to improve readability, generate figures and tables, and assist in drafting formal proofs. We did not use generative AI tools for any other task for which disclosure is required. We take full responsibility for the contents of this manuscript and have verified all AI-generated output, for example by cross-checking tables against the direct outputs of our execution scripts.

REFERENCES

Anthropic. Token counting. https://platform.claude.com/docs/en/build-with-claude/token-counting, 2026. Accessed July 2026.

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International conference on machine learning*, pp. 394–403. PMLR, 2018.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. Gpt-neox-20b: An open-source autoregressive language model. In *Proceedings of BigScience Episode# 5–Workshop on Challenges & Perspectives in Creating Large Language Models*, pp. 95–136, 2022.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and trends® in theoretical computer science*, 9(3-4):211–487, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pp. 265–284. Springer, 2006.

Adam Dziedzic, Haonan Duan, Muhammad Ahmad Kaleem, Nikita Dhawan, Jonas Guan, Yannis Cattan, Franziska Boenisch, and Nicolas Papernot. Dataset inference for self-supervised models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=CCBJf9xJo2X.

Google. Understand and count tokens. https://ai.google.dev/gemini-api/docs/tokens, 2026. Accessed July 2026.

Sivakanth Gopi, Pankaj Gulhane, Janardhan Kulkarni, Judy Hanwen Shen, Milad Shokouhi, and Sergey Yekhanin. Differentially private set union. In *International Conference on Machine Learning*, pp. 3627–3636. PMLR, 2020.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dirk Groeneveld, Iz Beltagy, Evan Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal

Nam, Matthew Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, William Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah Smith, and Hannaneh Hajishirzi. OLMo: Accelerating the science of language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15789–15809, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.841. URL https://aclanthology.org/2024.acl-long.841/.

Jonathan Hayase, Alisa Liu, Yejin Choi, Sewoong Oh, and Noah A. Smith. Data mixture inference attack: BPE tokenizers reveal training data compositions. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=EHXyeImux0.

Shlomo Hoory, Amir Feder, Avichai Tendler, Sofia Erell, Alon Peled-Cohen, Itay Laish, Hootan Nakhost, Uri Stemmer, Ayelet Benjamini, Avinatan Hassidim, et al. Learning and evaluating a differentially private pre-trained language model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 1178–1189, 2021.

Paul Jaccard. Étude comparative de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37(142):547–579, 1901. doi: 10.5169/seals-266450.

Brian Karrer, Daniel Kifer, Arjun Wilkins, and Danfeng Zhang. Exact privacy analysis of the gaussian sparse histogram mechanism. *Journal of Privacy and Confidentiality*, 14:1, 2024.

Kunho Kim, Sivakanth Gopi, Janardhan Kulkarni, and Sergey Yekhanin. Differentially private n-gram extraction. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=IQOawME4sqW.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, 2018.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 conference on empirical methods in natural language processing: System demonstrations*, pp. 66–71, 2018.

Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=hvdKKV2yt7T.

Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. LLM dataset inference: Did you train on my dataset? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=Fr9d1UMc37.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pp. 94–103, 2007. doi: 10.1109/FOCS.2007.66.

Matthieu Meeus, Shubham Jain, Marek Rei, and Yves-Alexandre de Montjoye. Did the neurons read your book? document-level membership inference for large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 2369–2385, 2024.

Jerzy Neyman and Egon Sharpe Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A*, 231(694-706): 289–337, 1933.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Natalia Ponomareva, Jasmijn Bastings, and Sergei Vassilvitskii. Training text-to-text transformers with privacy guarantees. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2182–2193, 2022.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026b. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Jessica Rumbelow and Matthew Watkins. Solidgoldmagikarp (plus, prompt generation). https://www.lesswrong.com/posts/aPeJE8bSo6rAFoLqg/solidgoldmagikarp-plus-prompt-generation, 2023. Accessed July 2026.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 1715–1725, 2016.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017.

Amer Sinha, Thomas Mesnard, Ryan McKenna, Daogao Liu, Christopher A Choquette-Choo, Yangsibo Huang, Da Yu, George Kaissis, Zachary Charles, Ruibo Liu, et al. Vaultgemma: A differentially private gemma model. *arXiv preprint arXiv:2510.15001*, 2025.

Chaofan Tao, Qian Liu, Longxu Dou, Niklas Muennighoff, Zhongwei Wan, Ping Luo, Min Lin, and Ngai Wong. Scaling laws with vocabulary: Larger models deserve larger vocabularies. *Advances in Neural Information Processing Systems*, 37:114147–114179, 2024.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Meng Tong, Yuntao Du, Kejiang Chen, Weiming Zhang, and Ninghui Li. Membership inference attacks on tokenizers of large language models. In *35th USENIX Security Symposium (USENIX Security 26)*, pp. 1207–1226, 2026.

Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the importance of difficulty calibration in membership inference attacks. In *International Conference on Learning Representations*, 2022.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *Proceedings of the 2022 ACM SIGSAC conference on computer and communications security*, pp. 3093–3106, 2022.

Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. In *International Conference on Machine Learning*, pp. 58244–58282. PMLR, 2024.

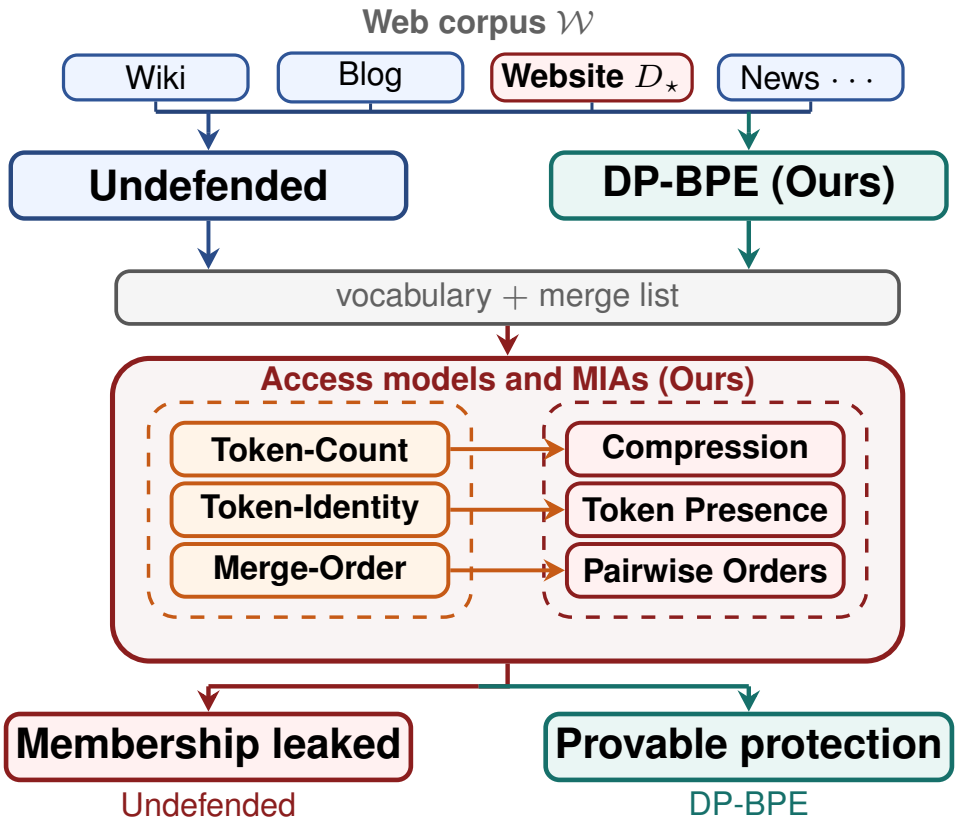

Figure 4: **Overview of our contributions.** We introduce three access models for MIAs against BPE tokenizers. Our attacks, one per access model, successfully infer membership of a target website $D_\star$ from undefended tokenizers. We also propose DP-BPE, a differentially private BPE training algorithm and the first to achieve website-level privacy guarantees.

A Diagram

Figure 4 gives an overview of the contributions Section 1 lists.

B Tokenizer Background

WordPiece (Wu et al., 2016) and the unigram language model of SentencePiece (Kudo, 2018; Kudo & Richardson, 2018) are the other common subword families, but BPE is the tokenizer released with modern LLMs (Grattafiori et al., 2024; Olmo et al., 2025; Team et al., 2025; Qwen Team, 2026a). BPE builds a subword vocabulary by starting from individual characters and repeatedly merging the most frequent adjacent pair in the training corpus (Sennrich et al., 2016). Algorithm 1 states the training loop. Each merge is the pair that most shortens the encoding of the corpus, so training is driven by compression and the resulting tokenizer is fitted to the text it saw. Therefore, the property our MIAs use holds, namely that training produces both a vocabulary and an ordered merge list whose order records which pairs were most frequent in the corpus at each step.

Vocabulary sizes of current LLMs. The vocabularies of released LLM tokenizers have grown from about 50,000 tokens in the GPT-NeoX-20B tokenizer, which Pythia and the first OLMo reuse (Black et al., 2022; Biderman et al., 2023; Groeneveld et al., 2024), to between 100,000 and 256,000 tokens in Llama 3 and the models released since 2025 (Table 3). Within model families, Llama 3 moved to 128,000 tokens to compress text better than the Llama 2 tokenizer (Grattafiori et al., 2024), and Qwen3.5 enlarged its vocabulary from about 150,000 to about 250,000 tokens (Qwen Team, 2026a), since fewer tokens per text lower the cost of training and inference. Scaling laws for the vocabulary suggest that the growth will continue, since Tao et al. (2024) find that most LLMs use vocabularies

Algorithm 1 Standard BPE training loop

Require: corpus D, base alphabet Σ_0, number of merges M
1: represent each training string as a sequence over Σ_0
2: initialize vocabulary $V \leftarrow \Sigma_0$ and merge list $\mathcal{M} \leftarrow \emptyset$
3: **for** $m = 1, \ldots, M$ **do**
4: count every adjacent pair (a, b) in the current corpus representation
5: choose a highest-count pair (a_m, b_m) using a fixed tie-breaking rule
6: add the new symbol $a_m b_m$ to V
7: append (a_m, b_m) to $\mathcal{M}$
8: replace each occurrence of adjacent (a_m, b_m) by $a_m b_m$ in the corpus representation
9: **end for**
10: output the tokenizer defined by $(V, \mathcal{M})$

Table 3: **The vocabularies of released LLM tokenizers keep growing.** Vocabulary size of the tokenizer released with each model, rounded to the nearest thousand, from the model's report or its released configuration.

Model	Year	Vocabulary size
GPT-NeoX-20B (Black et al., 2022)	2022	50,000
Pythia (Biderman et al., 2023)	2023	50,000
OLMo (Groeneveld et al., 2024)	2024	50,000
Llama 3 (Grattafiori et al., 2024)	2024	128,000
Olmo 3 (Olmo et al., 2025)	2025	100,000
Kimi-VL (Team et al., 2025)	2025	164,000
VaultGemma (Sinha et al., 2025)	2025	256,000
Qwen3.5 and Qwen3.6 (Qwen Team, 2026a;b)	2026	248,000
Llama 2 70B, compute-optimal (Tao et al., 2024)	2024	216,000

smaller than compute-optimal and predict that Llama 2 70B should have used at least about $216{,}000$ tokens, seven times its $32{,}000$ (Table 3).

C Attack Algorithms

Our three MIAs all instantiate Equation (3). For a candidate website w, the adversary extracts the statistic its access model allows from the target tokenizer and from the shadow tokenizers, fits IN and OUT reference models to the shadow statistics, and returns the log-likelihood ratio between them (Carlini et al., 2022). The three MIAs correspond to the three access models, namely *Token-Count*, *Token-Identity*, and *Merge-Order* access, and the Merge-Order Attack has two reference models, Bernoulli and Bradley-Terry. They differ only in the statistic $x(T, D_w)$ and in the distribution fitted to it, and Algorithms 2 to 4 state the resulting procedures.

Table 4: **Tokenizer access determines the attack surface.** Access models differ by whether the adversary observes only token counts, the emitted token identities, or the merge order. Each row is attacked by one of Appendices C.1 to C.3.

Access Model	**Observable**	**Observed Object**
Token-Count	$\mathcal{O}_{\text{count}}(T^\star) : \mathcal{X} \to \mathbb{N}$	Token count for queried text
Token-Identity	$\mathcal{O}_{\text{id}}(T^\star) : \mathcal{X} \to V(T^\star)^*$	Emitted token identities, or the vocabulary as an unordered set
Merge-Order	$\mathcal{O}_{\text{order}}(T^\star) = \Pi(T^\star)$	Merge list, or the vocabulary in the order training learned it

Algorithm 2 Token-Count Attack for one website

Require: target $T^\star$, website text D_w, shadows $\{T_i\}_{i=1}^N$ with $\mathcal{I}_w^{\text{in}}, \mathcal{I}_w^{\text{out}}$, variance floor η_{BPT}
1: compress D_w with the target: $b^\star \leftarrow \text{BPT}(T^\star, D_w)$ {Equation (4)}
2: compress D_w with each shadow: $b_i \leftarrow \text{BPT}(T_i, D_w)$
3: **for** $c \in \{\text{in}, \text{out}\}$ **do**
4: fit a Gaussian to $\{b_i\}_{i \in \mathcal{I}_w^c}$, giving $\widehat{\mu}_w^c$ and $(\widehat{\sigma}_w^c)^2$
5: clip: $(\widehat{\sigma}_w^c)^2 \leftarrow \max\{(\widehat{\sigma}_w^c)^2, \eta_{\text{BPT}}\}$
6: **end for**
7: **return** $\log \dfrac{\mathcal{N}\big(b^\star; \widehat{\mu}_w^{\text{in}}, (\widehat{\sigma}_w^{\text{in}})^2\big)}{\mathcal{N}\big(b^\star; \widehat{\mu}_w^{\text{out}}, (\widehat{\sigma}_w^{\text{out}})^2\big)}$

C.1 Token-Count Attack

The adversary observes only one number per query, the token count, so neither token identities, nor the vocabulary, nor the merge order enter this MIA at any point.

Statistic. The statistic is the scalar BPT of Equation (4),

$$x(T, D_w) = \text{BPT}(T, D_w) = \frac{\text{Bytes}(D_w)}{\text{TokenCount}(T(D_w))}.$$

The numerator is fixed by the website and the denominator is the only quantity that depends on the tokenizer, so this is a one-dimensional statistic and the IN and OUT references are one-dimensional too.

Reference model. We fit a Gaussian distribution $\widehat{\theta}_w^{\text{in}}$ to the BPT scores of the IN shadow tokenizers and a second Gaussian $\widehat{\theta}_w^{\text{out}}$ to those of the OUT shadow tokenizers. Each is fitted by the empirical mean of its group together with the empirical variance, clipped below by a public floor $\eta_{\text{BPT}} > 0$ so that a group whose shadow scores happen to coincide cannot produce an infinite log-likelihood ratio. Evaluating Equation (3) at $\text{BPT}(T^\star, D_w)$ gives

$$\begin{aligned} s_{\text{count}}(w) = &\log p\Big(\text{BPT}(T^\star, D_w) \mid \widehat{\theta}_w^{\text{in}}\Big) \\ &- \log p\Big(\text{BPT}(T^\star, D_w) \mid \widehat{\theta}_w^{\text{out}}\Big). \end{aligned}$$

C.2 Token-Identity Attack

The adversary obtains $V(T)$ without its order from the token identities the tokenizer emits or from the vocabulary as an unordered set, and this MIA treats the two alike since it uses nothing but membership in $V(T)$.

Statistic. The MIA considers only the tokens the website actually uses, namely

$$U_w = \{t : t \text{ appears when encoding } D_w \text{ under } T^\star\},$$

A tokenizer is then represented by which of these tokens its vocabulary contains,

$$x(T, D_w) = \big(\mathbf{1}[t \in V(T)]\big)_{t \in U_w},$$

since training on w makes exactly these tokens more likely to be learned.

Reference model. We model each token's presence as an independent Bernoulli variable. From the shadow tokenizers, we estimate the IN and OUT presence probabilities

$$q_t^c = \frac{\sum_{i \in \mathcal{I}_w^c} \mathbf{1}[t \in V(T_i)]}{|\mathcal{I}_w^c|}, \quad c \in \{\text{in}, \text{out}\}.$$

Filtering. The filter drops every token on which the two shadow groups agree, since such a token contributes a log-likelihood ratio near zero while still adding estimation noise. We measure the

Algorithm 3 Token-Identity Attack for one website

Require: target $T^\star$, website text D_w, shadows $\{T_i\}_{i=1}^N$ with $\mathcal{I}_w^{\text{in}}, \mathcal{I}_w^{\text{out}}$, smoothing α, separation measure d, threshold τ_{sep}, budget β

1: $U_w \leftarrow$ tokens emitted when $T^\star$ encodes D_w
2: **for** $t \in U_w$ **and** $c \in \{\text{in}, \text{out}\}$ **do**
3: estimate presence: $q_t^c \leftarrow \big(\sum_{i \in \mathcal{I}_w^c} \mathbf{1}[t \in V(T_i)] + \alpha\big) / \big(|\mathcal{I}_w^c| + 2\alpha\big)$
4: score the separation $\Delta_w(t) \leftarrow d\big(q_t^{\text{in}}, q_t^{\text{out}}\big)$
5: **end for**
6: $\widetilde{U}_w \leftarrow$ tokens with $\Delta_w(t) \geq \tau_{\text{sep}}$, keeping $\lfloor \beta |U_w| \rfloor$ of them
7: **return** $\displaystyle\sum_{t \in \widetilde{U}_w} \log \frac{q_t^{\text{in}}}{q_t^{\text{out}}}$

IN/OUT separation of a token by Cohen's h, the standard effect size for the gap between two proportions. A token is kept only if its separation,

$$\Delta_w(t) = \left| 2 \arcsin \sqrt{q_t^{\text{in}}} - 2 \arcsin \sqrt{q_t^{\text{out}}} \right|$$

is at least the threshold τ_{sep}. Cohen's h compares the two proportions after the transform $p \mapsto 2 \arcsin \sqrt{p}$, which stretches the ends of $[0, 1]$, so it treats a gap between 0.90 and 0.99 as larger than an equally wide gap around 0.5. This is what we want here, since a token the IN shadows almost always learn while the OUT shadows almost never do is the clearest evidence available. A budget of $\lfloor \beta |U_w| \rfloor$ tokens then caps how many are retained. The final token set for this MIA is

$$\widetilde{U}_w \subseteq U_w.$$

Score. Write $y_t = \mathbf{1}[t \in V(T^\star)]$ for the presence bit observed on the target tokenizer. The Bernoulli log-likelihood ratio score over the retained tokens is

$$s_{\text{id}}(w) = \sum_{t \in \widetilde{U}_w} \log \frac{\big(q_t^{\text{in}}\big)^{y_t} \big(1 - q_t^{\text{in}}\big)^{1 - y_t}}{\big(q_t^{\text{out}}\big)^{y_t} \big(1 - q_t^{\text{out}}\big)^{1 - y_t}}.$$

Every token of U_w is emitted when $T^\star$ encodes D_w, so it lies in $V(T^\star)$, and $\widetilde{U}_w \subseteq U_w$ then gives $y_t = 1$ for every retained token. Setting $y_t = 1$ leaves

$$s_{\text{id}}(w) = \sum_{t \in \widetilde{U}_w} \big(\log q_t^{\text{in}} - \log q_t^{\text{out}}\big).$$

The implementation scores a website with this sum.

C.3 MERGE-ORDER ATTACK

A merge list, or any equivalent ordered vocabulary, reveals the rank of every token. We do not score those ranks directly, since a rank is only meaningful relative to the vocabulary that contains it and that vocabulary differs between the target and each shadow tokenizer. The statistic is instead a comparison between pairs of tokens in U_w, which stays well defined whenever both tokens are present.

Statistic. Let U_w be the set of tokens the target tokenizer $T^\star$ emits when encoding D_w, listed as

$$U_w = (t_1, \ldots, t_{K_w}),$$

sorted by decreasing frequency within D_w.

The pair construction starts from the most frequent tokens of the website. Let $A_w = \min\{A_{\max}, K_w\}$, and take the first A_w tokens of U_w, the most frequent tokens of website w under $T^\star$. The candidate pair family is

$$\begin{aligned} P_w = &\{(t_i, t_j) : 1 \leq i < j \leq A_w\} \\ &\cup \{(t_i, t_j) : 1 \leq i \leq A_w,\ A_w < j \leq K_w\}. \end{aligned}$$

This keeps every pair among those A_w tokens and every pair of one of them with a remaining token, avoiding enumeration of all $\binom{K_w}{2}$ pairs while concentrating on frequent tokens from the candidate website w.

For tokenizer T, let $r_T(t)$ denote the rank of token t in the BPE order of T, where lower rank means earlier in the order. If a token from U_w is absent from a shadow tokenizer, we treat it as appearing after all present tokens, with deterministic tie-breaking among absent tokens.

A tokenizer is represented by how it orders the pairs in P_w,

$$x(T, D_w) = \big(\mathbf{1}[r_T(a) < r_T(b)]\big)_{(a,b)\in P_w},$$

where r_T denotes the merge-order rank under T, written r in the main text for $T = T^\star$. Including w in training moves earlier in the order the merges whose pair counts w increases, so these comparisons shift.

Reference model. We evaluate two variants, *Bernoulli* and *Bradley-Terry*. Both use the same order bits over the same retained pair set $\widetilde{P}_w$, and differ only in the distribution fitted to them.

Bernoulli Variant. We model each pair comparison as an independent Bernoulli variable. For each candidate pair $(a, b) \in P_w$, define the IN and OUT win counts

$$k_{ab}^{c} = \sum_{i\in\mathcal{I}_w^c} \mathbf{1}[r_{T_i}(a) < r_{T_i}(b)], \quad c \in \{\text{in}, \text{out}\},$$

with group sizes

$$n_{\text{in}} = |\mathcal{I}_w^{\text{in}}|, \qquad n_{\text{out}} = |\mathcal{I}_w^{\text{out}}|.$$

With additive smoothing parameter $\alpha \geq 0$, define

$$q_{ab}^{\text{in}} = \frac{k_{ab}^{\text{in}} + \alpha}{n_{\text{in}} + 2\alpha}, \qquad q_{ab}^{\text{out}} = \frac{k_{ab}^{\text{out}} + \alpha}{n_{\text{out}} + 2\alpha}.$$

We retain only pairs whose IN and OUT order probabilities are well separated, measuring the separation by the Hellinger distance between the two Bernoullis,

$$\Delta_w(a,b)^2 = \tfrac{1}{2}\Big[\big(\sqrt{q_{ab}^{\text{in}}} - \sqrt{q_{ab}^{\text{out}}}\big)^2 + \big(\sqrt{1-q_{ab}^{\text{in}}} - \sqrt{1-q_{ab}^{\text{out}}}\big)^2\Big].$$

Like Cohen's h above, it separates probabilities near 0 and 1 more strongly than their raw difference does, and it is bounded in $[0, 1]$, so one badly estimated pair cannot dominate the ranking. Pairs with separation below τ_{sep} are discarded. A budget of $\lfloor\beta|U_w|\rfloor$ pairs caps how many are retained. The final pair set is

$$\widetilde{P}_w \subseteq P_w.$$

Write $y_{ab} = \mathbf{1}[r_{T^\star}(a) < r_{T^\star}(b)]$ for the order bit observed on the target tokenizer. Under a conditional-independence approximation across retained pairs, the Bernoulli score is

$$s_{\text{order-Bern}}(w) = \sum_{(a,b)\in\widetilde{P}_w} \log \frac{\big(q_{ab}^{\text{in}}\big)^{y_{ab}}\big(1-q_{ab}^{\text{in}}\big)^{1-y_{ab}}}{\big(q_{ab}^{\text{out}}\big)^{y_{ab}}\big(1-q_{ab}^{\text{out}}\big)^{1-y_{ab}}}.$$

This is the Merge-Order variant we report in the main experiments.

Bradley-Terry Variant. Ranks form one total order, so the pair comparisons are transitively constrained, which the independent Bernoulli parameters of the previous variant leave unused. Bradley-Terry uses this constraint by replacing them with a shared latent ranking model in which every comparison a token takes part in informs the same strength parameter (Bradley & Terry, 1952).

For Bradley-Terry fitting, we skip shadow tokenizers in which both tokens of a retained pair are absent. Among the remaining shadows, we count how often a appears before b and how often b

Algorithm 4 Merge-Order Attack (Bernoulli) for one website

Require: target $T^\star$, website text D_w, shadows $\{T_i\}_{i=1}^N$ with $\mathcal{I}_w^{\text{in}}, \mathcal{I}_w^{\text{out}}$, smoothing α, separation measure d, threshold τ_{sep}, budget β, cap $A_{\max}$

$U_w = (t_1, \ldots, t_{K_w}) \leftarrow$ tokens emitted when $T^\star$ encodes D_w, by decreasing frequency
keep the $A_w = \min\{A_{\max}, K_w\}$ most frequent tokens
$P_w \leftarrow$ every pair among them and every pair of one of them with a remaining token
for $(a, b) \in P_w$ **and** $c \in \{\text{in}, \text{out}\}$ **do**
    estimate order: $q_{ab}^c \leftarrow \big(\sum_{i \in \mathcal{I}_w^c} \mathbf{1}[r_{T_i}(a) < r_{T_i}(b)] + \alpha\big) / \big(|\mathcal{I}_w^c| + 2\alpha\big)$
    score the separation $\Delta_w(a, b) \leftarrow d\big(q_{ab}^{\text{in}}, q_{ab}^{\text{out}}\big)$
end for
$\widetilde{P}_w \leftarrow$ pairs with $\Delta_w(a, b) \geq \tau_{\text{sep}}$, keeping $\lfloor \beta |U_w| \rfloor$ of them
extract the order bits from the target: $y_{ab} \leftarrow \mathbf{1}[r_{T^\star}(a) < r_{T^\star}(b)]$
return $\displaystyle\sum_{(a,b) \in \widetilde{P}_w} \log \frac{(q_{ab}^{\text{in}})^{y_{ab}} (1 - q_{ab}^{\text{in}})^{1 - y_{ab}}}{(q_{ab}^{\text{out}})^{y_{ab}} (1 - q_{ab}^{\text{out}})^{1 - y_{ab}}}$

appears before a, separately for IN and OUT shadows. Denote these directed counts by $c_{a \succ b}^{\text{in}}$ and $c_{b \succ a}^{\text{in}}$, and analogously for OUT. Each direction is smoothed by a pseudo-win $\lambda_{\text{BT}} > 0$ added before fitting, so that a pair one side never reverses does not force an unbounded strength difference.

The Bradley-Terry model associates each token t appearing in $\widetilde{P}_w$ with a latent strength parameter θ_t and defines

$$\Pr[a \succ b] = \text{logistic}(\theta_a - \theta_b),$$

where $a \succ b$ means $r_T(a) < r_T(b)$ and $\text{logistic}(x) = 1/(1 + \exp(-x))$. Thus all comparisons involving the same token share one latent parameter. The model therefore needs no comparison of every token pair and summarizes many pairwise observations with one parameter per token.

We fit one Bradley-Terry model to the IN win counts and one to the OUT win counts. Let $\boldsymbol{\theta}_w^{\text{in}}$ and $\boldsymbol{\theta}_w^{\text{out}}$ denote the fitted strength vectors. They imply fitted pairwise probabilities

$$q_{ab}^{\text{in}} = \text{logistic}(\theta_a^{\text{in}} - \theta_b^{\text{in}}), \qquad q_{ab}^{\text{out}} = \text{logistic}(\theta_a^{\text{out}} - \theta_b^{\text{out}}).$$

We fit each model by maximizing its log-likelihood, and clip the fitted probabilities away from 0 and 1 before taking log ratios.

The score has the same form as in the Bernoulli variant, with the fitted probabilities substituted for the empirical ones,

$$s_{\text{order-Bradley}}(w) = \sum_{(a,b) \in \widetilde{P}_w} \log \frac{\big(q_{ab}^{\text{in}}\big)^{y_{ab}} \big(1 - q_{ab}^{\text{in}}\big)^{1 - y_{ab}}}{\big(q_{ab}^{\text{out}}\big)^{y_{ab}} \big(1 - q_{ab}^{\text{out}}\big)^{1 - y_{ab}}}.$$

The two Merge-Order variants therefore share the access model, the retained pair set, and the score, and differ only in how the pair probabilities are estimated, with independent per-pair frequencies replaced by a shared latent ranking model.

C.4 OFFLINE VARIANTS

The MIAs above fit an IN and an OUT reference model from shadow tokenizers trained with and without the candidate website. In the offline setting, the same as that of LiRA (Carlini et al., 2022), the adversary uses only shadow tokenizers trained without the candidate website, so it can fit only the OUT reference model. The goal is an MIA whose shadow tokenizers can be trained once, before the candidate websites are known, and reused for all of them, since none needs to contain a candidate. In place of the log-likelihood ratio of Equation (3), each offline MIA scores how unlikely the target's statistic is under the OUT reference model. We describe the offline variants of the Token-Count Attack, the Token-Identity Attack and Merge-Order with Bernoulli by how each differs from its online counterpart.

Token-Count Attack. We fit only the OUT Gaussian of Algorithm 2, and the offline score is the standardized distance of the target's BPT from it,

$$s^{\mathrm{off}}_{\mathrm{count}}(w) = \frac{\mathrm{BPT}(T^\star, D_w) - \widehat{\mu}^{\mathrm{out}}_w}{\widehat{\sigma}^{\mathrm{out}}_w},$$

which is large for a member website, since the target compresses it better than the OUT shadow tokenizers do.

Token-Identity Attack and Merge-Order with Bernoulli. We write F_w for U_w or P_w, and y_f for the bit the target shows on $f \in F_w$, namely $\mathbf{1}[t \in V(T^\star)]$ or $\mathbf{1}[r_{T^\star}(a) < r_{T^\star}(b)]$. Offline, only the OUT probability q^{out}_f of each bit is estimated.

Filtering. Lacking the IN group, the offline filter keeps an element only if the confidence $|q^{\mathrm{out}}_f - 1/2|$ of the OUT reference model is at least the threshold τ_{out}, since a bit whose OUT probability is near $1/2$ is almost equally likely either way and carries little evidence. The final set $\widetilde{F}_w \subseteq F_w$ takes the place of $\widetilde{U}_w$ or $\widetilde{P}_w$.

Score. The log-likelihood of the target under the OUT reference model is

$$L_w = \sum_{f \in \widetilde{F}_w} \Big[y_f \log q^{\mathrm{out}}_f + (1 - y_f) \log \big(1 - q^{\mathrm{out}}_f\big) \Big].$$

Under a conditional-independence approximation across retained elements, each y_f is a Bernoulli variable with parameter q^{out}_f under the OUT reference model, so L_w has mean and variance

$$\mu_w = \sum_{f \in \widetilde{F}_w} \Big[q^{\mathrm{out}}_f \log q^{\mathrm{out}}_f + \big(1 - q^{\mathrm{out}}_f\big) \log \big(1 - q^{\mathrm{out}}_f\big) \Big],$$

$$\sigma^2_w = \sum_{f \in \widetilde{F}_w} q^{\mathrm{out}}_f \big(1 - q^{\mathrm{out}}_f\big) \Big(\log \frac{q^{\mathrm{out}}_f}{1 - q^{\mathrm{out}}_f} \Big)^2.$$

The offline score is the negated standardized log-likelihood

$$s^{\mathrm{off}}(w) = - \frac{L_w - \mu_w}{\sigma_w},$$

negated because the target is less likely under the OUT reference model when w is a member. Without standardizing, the score would grow with $|\widetilde{F}_w|$, which varies by orders of magnitude across websites, and would rank websites by size rather than by membership.

D DP-BPE ALGORITHMIC DETAILS

We give DP-BPE in full, with the calibration of its noise scale and threshold and the proof of its guarantee, keeping the step markers ❶–❻ of Figure 2 throughout.

In summary, DP-BPE releases one DP word histogram of the training corpus ❶–❺ and then trains a BPE tokenizer on it ❻. The histogram is released once under a fixed privacy budget (ε, δ) with the website as the unit of protection, and every later step depends only on that histogram, so the vocabulary and merge list inherit the same guarantee at no additional cost.

Notation. Let $\mathcal{D}$ denote the training corpus, a set of websites w, where the website is our unit of privacy, and let two corpora be adjacent if one is obtained from the other by adding or removing a single website. We write $\mathcal{U}$ for the universe of words the pre-tokenizer produces, which is unbounded and depends on the corpus, and Φ for the standard normal cumulative distribution function.

D.1 PER-WEBSITE PRE-TOKENIZATION AND CLIPPING

Pre-tokenization. DP-BPE first applies a fixed, data-independent pre-tokenizer that segments each website w ❶ into words. Because it is public and applied identically to every website, this step

Algorithm 5 Per-website clipping $\mathrm{Clip}(w)$

Require: website w, public pre-tokenizer $\mathcal{P}$, word cap $N_{\max}$, norm bound $L_{\max}$
Ensure: contribution v_w^{clip} with $\|v_w^{\mathrm{clip}}\|_0 \le N_{\max}$ and $\|v_w^{\mathrm{clip}}\|_2 \le L_{\max}$
$\mathcal{W}_w \leftarrow$ set of distinct words appearing in $\mathcal{P}(w)$ {presence only; repetitions discarded}
if $|\mathcal{W}_w| > N_{\max}$ **then**
  $S_w \leftarrow$ uniformly random subset of $\mathcal{W}_w$ of size $N_{\max}$ {word cap}
else
  $S_w \leftarrow \mathcal{W}_w$
end if
$v_{w,j}^0 \leftarrow \mathbf{1}[j \in S_w]$ for all j
$v_w^{\mathrm{clip}} \leftarrow v_w^0 \cdot \min\{1,\, L_{\max}/\|v_w^0\|_2\}$ {norm bound}
return v_w^{clip}

consumes no privacy budget. We represent each website by the *set* of words it contains ❷, that is, by the binary vector $v_w \in \{0,1\}^{\mathcal{U}}$ with

$$v_{w,j} \;=\; \mathbf{1}[\text{word } j \text{ occurs in website } w]\,.$$

Repetition inside a website is deliberately discarded, so that the aggregate $\sum_w v_w$ counts for each word how many websites contain it, a *website frequency*. This bounds the influence of one website on each word count by 1 without any further count clipping.

Per-website clipping. One quantity is still unbounded, namely the number of words a website may contribute. A website may introduce arbitrarily many unseen words, so the release must be restricted to observed words. DP-BPE therefore bounds each contribution in two further stages.

(i) Word cap ❸. Each website may contribute at most $N_{\max}$ words. If $\|v_w\|_0 > N_{\max}$, we keep a uniformly random subset S_w of the words it contains with $|S_w| = N_{\max}$ and zero the rest,

$$v_{w,j}^0 \;=\; v_{w,j} \cdot \mathbf{1}[j \in S_w]\,.$$

The subsets S_w are drawn independently across websites and depend only on that website's own data, so the subsampling does not affect the noise scale or the threshold. The cap limits how many new words a single website can introduce, which is what the threshold calibration below rests on.

(ii) Norm bound ❹. The result is then rescaled to Euclidean norm at most $L_{\max}$,

$$v_w^{\mathrm{clip}} \;=\; v_w^0 \cdot \min\left\{1,\; \frac{L_{\max}}{\|v_w^0\|_2}\right\},$$

so that adjacent corpora differ in the aggregate by a single vector with Euclidean norm at most $L_{\max}$ and at most $N_{\max}$ non-zero entries. The word cap alone already bounds this norm, since v_w^0 is binary with at most $N_{\max}$ non-zero entries and therefore satisfies

$$\|v_w^0\|_2 \;=\; \sqrt{\|v_w^0\|_0} \;\le\; \sqrt{N_{\max}}.$$

Choosing $L_{\max} < \sqrt{N_{\max}}$ is what sets the sensitivity to $L_{\max}$ rather than leaving it at the $\sqrt{N_{\max}}$ the cap gives on its own. Algorithm 5 summarizes the procedure.

The norm bound is what keeps the noise scale independent of the word cap. Without it, the noise would have to be calibrated to $\sqrt{N_{\max}}$, a bound that is tight only for a website that actually realizes all $N_{\max}$ permitted words. Most websites contain far fewer, yet the noise would be calibrated to the largest one, so a large word cap would raise σ and τ together until no word exceeded the release threshold. The explicit bound instead fixes the calibration at $L_{\max}$ however $N_{\max}$ is set. Rescaling multiplies a website's whole vector by one factor, so v_w^{clip} still marks exactly which words that website contains, and only the weight the site carries in the aggregate changes.

D.2 Privatizing the Word Counts

Given the clipped contributions, DP-BPE releases the aggregate word histogram ❺ using the Gaussian Sparse Histogram Mechanism (Karrer et al., 2024), which is designed for high-dimensional histograms with a small number of non-zero entries. It operates on the clean aggregate $a = \sum_{w \in \mathcal{D}} v_w^{\mathrm{clip}}$

and tests only the words that at least one website contributes. Each tested word receives independent Gaussian noise, and a word is released only if its noisy count exceeds a threshold τ,

$$\tilde{a}_j \;=\; a_j + \xi_j, \quad \xi_j \sim \mathcal{N}(0, \sigma^2), \quad \mathcal{S} \;=\; \{\, j : \tilde{a}_j > \tau \,\}.$$

The noise protects the *counts* of words that appear in both neighboring corpora, while the threshold protects *which words are released*, since a word present in one corpus and absent from the other would otherwise be revealed by its appearance in the output alone. The privacy budget is split as $\delta = \delta_{\mathrm{sel}} + \delta_{\mathrm{cnt}}$ across the two failure modes.

Noise calibration. We choose σ with the analytic Gaussian mechanism (Balle & Wang, 2018), which gives the smallest σ such that releasing a, which changes by a single vector of Euclidean norm at most $L_{\max}$ between adjacent corpora, is $(\varepsilon, \delta_{\mathrm{cnt}})$-DP.

Threshold calibration. Consider a word carried by a single website, so that it is absent from the neighboring corpus and never tested there. Its clean count is at most 1, because every entry starts at 1 and the rescaling can only shrink it. Following Karrer et al. (2024), setting

$$\tau \;=\; 1 \;+\; \sigma\, \Phi^{-1}\!\left(1 - \frac{\delta_{\mathrm{sel}}}{N_{\max}}\right)$$

makes the probability that any *one* such word is released at most $\delta_{\mathrm{sel}}/N_{\max}$. Since the word cap guarantees that a single website introduces at most $N_{\max}$ new words, a union bound keeps the probability that any of them is revealed below δ_{sel}. Without a finite $N_{\max}$ the union bound, and hence the threshold, would be undefined.

Released counts and rounding. For each released word, DP-BPE reuses the noisy value $\tilde{a}_j$ as the released count rather than drawing fresh noise, and rounds it to the nearest integer, floored at one,

$$\hat{a}_j \;=\; \max\{1,\ \lceil \tilde{a}_j \rfloor\}\,,$$

so that every released word carries a positive integer count. Both steps are post-processing of the noisy counts and cost no additional privacy budget. This yields the integer histogram $\hat{H} = \{(j, \hat{a}_j) : j \in \mathcal{S}\}$ expected by a standard BPE trainer. Algorithm 6 states the full release.

Combining the two failure modes gives the formal version of Theorem 1, stated for the tokenizer that Algorithm 7 releases.

Theorem 2 (Website-level privacy of DP-BPE)**.** *Let $\varepsilon > 0$ and $\delta_{\mathrm{sel}}, \delta_{\mathrm{cnt}} > 0$, and let $N_{\max}, L_{\max} < \infty$. Run Algorithm 7 with the noise scale σ returned by the analytic Gaussian mechanism (Balle & Wang, 2018) at budget $(\varepsilon, \delta_{\mathrm{cnt}})$ and Euclidean sensitivity $L_{\max}$, and with threshold $\tau = 1 + \sigma\,\Phi^{-1}(1 - \delta_{\mathrm{sel}}/N_{\max})$. Then, for any number of merges M, the released vocabulary and merge list $(\mathcal{V}, \mathcal{M})$ are $(\varepsilon, \delta_{\mathrm{sel}} + \delta_{\mathrm{cnt}})$-DP with respect to the addition or removal of one website.*

Proof. Let $\mathcal{D}' = \mathcal{D} \cup \{w^\star\}$ be adjacent to $\mathcal{D}$, and write

$$a = \sum_{w \in \mathcal{D}} v_w^{\mathrm{clip}}, \qquad a' = \sum_{w \in \mathcal{D}'} v_w^{\mathrm{clip}}$$

for the two aggregates Algorithm 6 releases. Each website is clipped on its own, so $a' - a = v_{w^\star}^{\mathrm{clip}}$, which by Algorithm 5 has at most $N_{\max}$ non-zero entries and Euclidean norm at most $L_{\max}$.

Consider first the words with a positive count under both aggregates, so both runs test them. Restricting $a' - a$ to them leaves its norm at most $L_{\max}$, so on these words the release is the Gaussian mechanism at sensitivity $L_{\max}$, which is $(\varepsilon, \delta_{\mathrm{cnt}})$-DP by the calibration of σ (Balle & Wang, 2018).

Consider next the words observed under one corpus only, which the other run never tests. Each such word is carried by $w^\star$ alone, so its clean count is at most 1, since every entry of $v_{w^\star}$ starts at 1 and the rescaling can only shrink it. Such a word is released only if $\tilde{a}_j > \tau$, an event of probability at most $1 - \Phi((\tau - 1)/\sigma) = \delta_{\mathrm{sel}}/N_{\max}$ by the choice of τ. There are at most $N_{\max}$ such words, so a union bound bounds the probability that any of them is released by δ_{sel}. Conditioned on this event not occurring, the two runs test the same set of words, and the previous paragraph applies.

The two failure modes are disjoint and their budgets add, so the histogram $\hat{H}$ is $(\varepsilon, \delta_{\mathrm{sel}} + \delta_{\mathrm{cnt}})$-DP. Rounding the released counts depends only on $\hat{H}$ and does not weaken this. Algorithm 7 computes $(\mathcal{V}, \mathcal{M})$ from $\hat{H}$ alone, for any M, and a function of a DP output is DP with the same parameters. □

Algorithm 6 DP word histogram release

Require: corpus $\mathcal{D}$, word cap $N_{\max}$, norm bound $L_{\max}$, privacy budget (ε, δ), split ratio $\rho \in (0, 1)$
Ensure: (ε, δ)-DP integer word histogram $\hat{H}$

$\delta_{\text{sel}} \leftarrow \rho\,\delta$; $\quad \delta_{\text{cnt}} \leftarrow (1-\rho)\,\delta$
$\sigma \leftarrow \textsc{AnalyticGaussian}(\varepsilon, \delta_{\text{cnt}}, L_{\max})$
$\tau \leftarrow 1 + \sigma\,\Phi^{-1}(1 - \delta_{\text{sel}}/N_{\max})$
for all websites $w \in \mathcal{D}$ **do**
  $v_w^{\text{clip}} \leftarrow \text{Clip}(w)$ {Algorithm 5}
end for
$a \leftarrow \sum_{w \in \mathcal{D}} v_w^{\text{clip}}$; $\quad \hat{H} \leftarrow \emptyset$
for all words j with $a_j > 0$ **do** {observed words only}
  $\tilde{a}_j \leftarrow a_j + \xi_j, \;\; \xi_j \sim \mathcal{N}(0, \sigma^2)$
  if $\tilde{a}_j > \tau$ **then**
    $\hat{H} \leftarrow \hat{H} \cup \{(j,\ \max\{1, \lceil \tilde{a}_j \rfloor\})\}$ {reuse noisy value; round}
  end if
end for
return $\hat{H}$

Algorithm 7 DP-BPE

Require: corpus $\mathcal{D}$, word cap $N_{\max}$, norm bound $L_{\max}$, privacy budget (ε, δ), split ratio ρ, number of merges M
Ensure: (ε, δ)-DP BPE tokenizer $(\mathcal{V}, \mathcal{M})$

$\hat{H} \leftarrow \textsc{DpHistogram}(\mathcal{D}, N_{\max}, L_{\max}, \varepsilon, \delta, \rho)$ {Algorithm 6; the only private step}
$(\mathcal{V}, \mathcal{M}) \leftarrow$ standard BPE training on $\hat{H}$ with M merges {Algorithm 1, post-processing}
return $(\mathcal{V}, \mathcal{M})$

D.3 Tokenizer Training as Post-Processing

The released histogram $\hat{H}$ is the only object that depends on the private corpus. DP-BPE therefore runs the standard BPE training loop of Algorithm 1 on $\hat{H}$ ❻, treating each released pair $(j, \hat{a}_j)$ as a word j observed $\hat{a}_j$ times. The trainer is unmodified, and Algorithm 7 states the procedure.

Because this stage depends on $\mathcal{D}$ only through $\hat{H}$, it is post-processing of an (ε, δ)-DP release, so the released vocabulary $\mathcal{V}$ and ordered merge list $\mathcal{M}$ inherit the guarantee at no additional cost, whatever the target vocabulary size. No part of the privacy budget is spent on the merge search, however many merges are performed.

E Baselines

We compare against the attack and defense baselines of Tong et al. (2026), described in full below.

E.1 Attack Baselines

We include four attack baselines adapted from prior tokenizer attacks (Tong et al., 2026), aligning each baseline with the corresponding access model in our setup.

Compression Rate. Under *Token-Count* access, we evaluate against Tong et al. (2026)'s BPT compression signal, which computes the compression rate of the tokenizer on the target website. This follows directly from the goal of the tokenizer, which is to maximize the compression rate of the text, so a larger compression rate suggests that the website was part of the training data.

Vocabulary Overlap. Under *Token-Identity* access, we adapt their distinctive-token overlap attack, which uses shadow tokenizers trained with and without the candidate website to identify distinctive tokens, *i.e.,* tokens that appear only when the candidate website is included, and then measures their Jaccard overlap (Jaccard, 1901) with the target vocabulary.

Frequency. Under *Merge-Order* access, we adapt their frequency estimation attack, which uses a shadow tokenizer to calibrate the relationship between token merge order and token frequency. Their attack then scores a candidate website using a metric called Relative Token Frequency with Self-information (RTF-SI) that combines how concentrated a token is in the candidate website with how rare the token is estimated to be from its merge order.

Naive Bayes. Also under *Merge-Order* access, we adapt their rare token probabilistic baseline, which uses shadow tokenizers to estimate a per-token membership signal, selects the top k tokens that appear latest in the merge order, and combines their evidence using the standard Naive Bayes approach, which assumes conditional independence between tokens.

All baselines, apart from Compression Rate, which operates without shadow tokenizers, use the same shadow split at the website level as our attacks for fair evaluation.

E.2 Defense Baselines

We compare DP-BPE against the two defense baselines of Tong et al. (2026), one applied to the tokenizer after standard BPE training and one that changes the merge search itself.

Min-Count filtering. We evaluate the Min-Count post-processing defense from (Tong et al., 2026). After training a standard BPE tokenizer, the defender removes any learned token whose aggregate count in the tokenizer training corpus is below a public threshold $n_{\min}$, together with the corresponding merge entry. This filters rare, weakly supported tokens that are likely to be dataset-specific, but it does not change the BPE training trajectory or the order the corpus imposed on the remaining merges. The standard BPE tokenizer without this filtering defense corresponds to $n_{\min} = 0$.

Softmax BPE. We additionally consider the Softmax BPE defense of Tong et al. (2026), which is inspired by the exponential mechanism (McSherry & Talwar, 2007) for DP. At each merge step, Algorithm 1 takes the adjacent pair of highest count. Softmax BPE instead draws it, writing $\mathcal{C}$ for the candidate pairs of the current corpus representation and $c(a, b)$ for the count of pair (a, b),

$$\Pr[(a,b) \text{ merged}] = \frac{\exp\big(\lambda\, c(a,b)\big)}{\sum_{(a',b')\in\mathcal{C}} \exp\big(\lambda\, c(a',b')\big)}.$$

The parameter λ interpolates between the two extremes, since $\lambda \to \infty$ recovers standard BPE and $\lambda \to 0$ picks a merge uniformly at random. The exponential mechanism (McSherry & Talwar, 2007) would set $\lambda = \varepsilon/(2\Delta)$ for a utility function of sensitivity Δ, but one website can contribute arbitrarily many occurrences of a pair, so Δ is unbounded here and no finite λ corresponds to a finite ε.

F Experimental Details

Dataset. Tokenizers in real-world LLMs are trained on web content, so we evaluate on the C4 corpus (Raffel et al., 2020), a large-scale dataset of cleaned and filtered web pages from Common Crawl. We sample 50 million web pages from the English training split of C4, extract the host name from each page's URL, and retain only websites with at least 200 pages, yielding $11{,}233{,}234$ web pages across $21{,}090$ websites. This source universe is larger than the C4 tokenizer-MIA benchmark of Tong et al. (2026), which uses $1{,}681{,}296$ web pages across $4{,}133$ websites. Each website is treated as an individual dataset $D_w \in \mathcal{D}$. For the tokenizer experiments, we subsample 50 pages from each website, drawn uniformly at random. Target and shadow tokenizers are trained on websites drawn from this same universe of $21{,}090$ websites, and every website serves as a candidate. Taking the same number of pages from every website also removes a membership signal that would come from website size alone. After subsampling, the corpus contains 2.21 GB of UTF-8 text and $366{,}251{,}784$ words when split on whitespace. A website thus contributes on average $104{,}892$ bytes, or $2{,}098$ bytes per page.

Tokenizers. All experiments use the same six vocabulary sizes,

$$|\mathcal{V}| \in \{40{,}000,\ 80{,}000,\ 110{,}000,\\ 140{,}000,\ 170{,}000,\ 200{,}000\}.$$

For each setting, we train 10 target tokenizers and 128 shadow tokenizers on website-level corpora from the same website universe, using the same preprocessing pipeline throughout. Each tokenizer

trains on a uniformly random half of the $21{,}090$ websites, $10{,}545$ of them, drawn under its own seed, so a candidate website is a member of any given shadow tokenizer with probability $1/2$ and of about half the shadow tokenizers overall. For a fixed candidate, the shadow tokenizers that include it form the IN set and those that exclude it form the OUT set. Unless we state otherwise, our MIAs fit their IN and OUT reference models from shadow tokenizers trained under the same setting as the target.

Metrics. Aggregate tables report the mean over target tokenizers with its standard error.

Compute. All tokenizer training and all MIAs run on CPU. Training one BPE tokenizer with a vocabulary of $200{,}000$ tokens on $10{,}545$ websites takes around 3 minutes on a 2×56-core Xeon Gold 5420+ CPU, so the 128 shadow tokenizers used throughout our evaluation cost under 7 hours of serial compute.

Min-Count. The defense filters the exported tokenizer after standard BPE training, as Appendix E.2 describes, so we reuse one unconstrained training run per corpus across all thresholds and tokenizers that differ only in $n_{\min}$ differ only in which learned tokens and merge entries survive. We evaluate $n_{\min} \in \{32, 64, 128, 256\}$.

DP-BPE. We use the word cap $N_{\max} = 50{,}000$ and the norm bound $L_{\max} = 30$ of Appendix D.1 in all reported runs, and evaluate the privacy budgets $\varepsilon \in \{1, 3, 10\}$ at a fixed $\delta = 5 \times 10^{-5}$, split by $\rho = 0.5$ into $\delta_{\text{sel}} = \delta_{\text{cnt}} = 2.5 \times 10^{-5}$.

Softmax BPE. We evaluate the leakage this defense admits and the utility it costs at $\lambda \in \{1, 10, 20\}$, in both cases against targets at a vocabulary of $200{,}000$, and the leakage at $\lambda = 20$ also at the further vocabulary sizes of Table 20, where λ is the sharpness parameter of Appendix E.2.

G Additional Attack Results

G.1 Full Undefended Results

The undefended leakage is not confined to the false positive rate we report at. Figure 5 gives the undefended ROC curves for every attack at each of the six target vocabulary sizes. Within every access model, our strongest MIA leads every prior baseline at 1% FPR and above at every vocabulary size, so our conclusions do not depend on the 1% FPR threshold we report at. At 0.1% FPR and below, Vocabulary Overlap can lead our Token-Identity Attack at the two largest vocabularies. At the extreme left of the curves, which of our three MIAs leads depends on the false positive rate, and only at a vocabulary of $200{,}000$ does Token-Count lead throughout. Table 5 gives the ROC-AUC of every attack at every vocabulary size, where our MIAs also lead the prior MIAs under each access model.

Websites differ so widely in how compactly they tokenize that no single threshold could infer membership from a token count, which is what makes the comparison against shadow tokenizers necessary under *Token-Count* access. Figure 6 shows raw BPT distributions for nine websites at a vocabulary of $80{,}000$, whose characteristic values span roughly 3.8 to 6.0, a range far wider than the shift membership itself produces. Comparing each website against its own IN and OUT shadow tokenizers removes that spread, which makes compression an informative membership signal in Table 1.

G.2 Smaller Vocabularies

Leakage is still present at vocabulary sizes below $40{,}000$. We train undefended target and shadow tokenizers at $10{,}000$, $15{,}000$, $20{,}000$ and $30{,}000$ tokens on the same websites and splits as the main experiments (Table 6). At $10{,}000$ the Token-Identity Attack and Merge-Order with Bernoulli both reach 4.8% TPR@1%FPR, and the Token-Count Attack 1.1%, which is no better than guessing. The lead of Merge-Order with Bernoulli over the Token-Identity Attack shrinks with the vocabulary, from 4.8 points at $40{,}000$ to 3.4 at $30{,}000$ and to none at $10{,}000$, and by ROC-AUC the Token-Identity Attack leads Merge-Order with Bernoulli at every size below $40{,}000$ (Table 7). Together with Table 1, every one of our MIAs grows with the vocabulary over the whole range from $10{,}000$ to $200{,}000$.

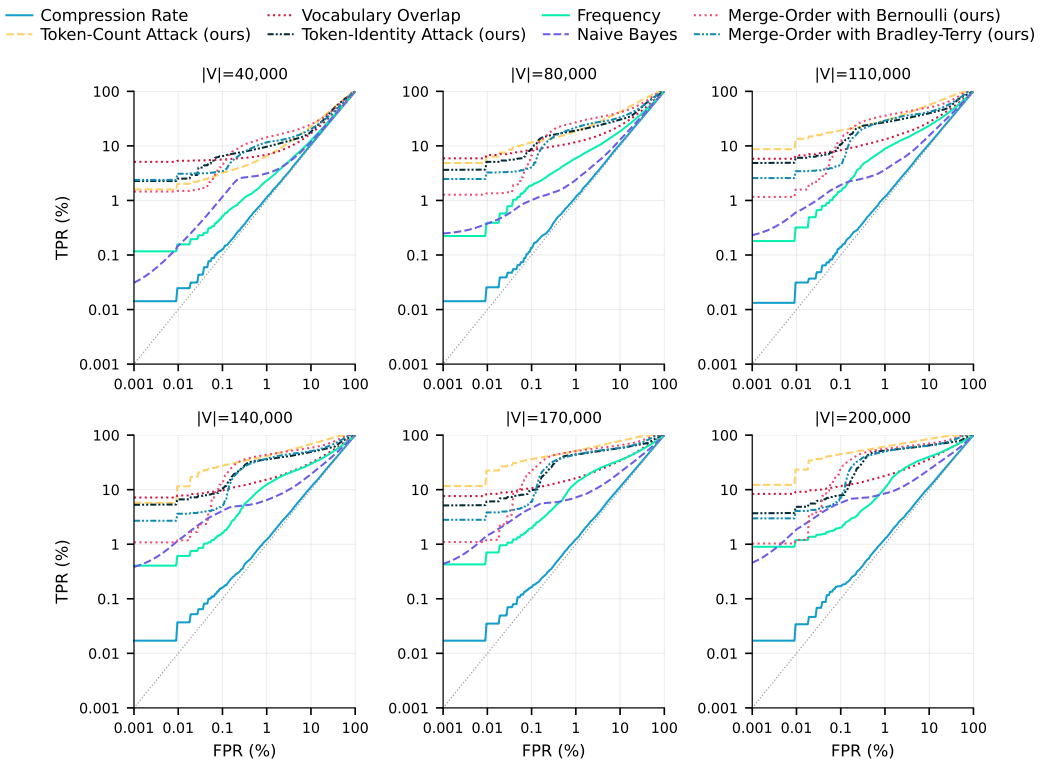

Figure 5: **Our attacks outperform the prior baselines from** 1% **FPR upward.** Log-log ROC curves for every attack against the undefended tokenizer, one line per attack and one panel per target vocabulary size. The baselines are those of Tong et al. (2026).

Table 5: **Our attacks lead the prior baselines by ROC-AUC as well as at** 1% **FPR.** ROC-AUC (%) for every attack against the undefended tokenizer at every vocabulary size. Bold marks the strongest attack in each column within its access model. ROC-AUC counterpart of Table 1.

Access model	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
		40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	Compression Rate (Tong et al., 2026)	50.3±0.1	50.7±0.1	50.9±0.1	51.2±0.1	51.3±0.1	51.5±0.1
	Token-Count Attack (ours)	**63.6±0.2**	**77.8±0.1**	**85.2±0.1**	**90.5±0.1**	**93.7±0.1**	**95.8±0.0**
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	55.2±0.1	59.7±0.1	62.4±0.1	65.4±0.2	67.4±0.2	69.3±0.2
	Token-Identity Attack (ours)	**60.5±0.3**	**65.8±0.1**	**69.8±0.1**	**73.8±0.1**	**77.5±0.1**	**80.9±0.1**
Merge-Order	Frequency (Tong et al., 2026)	52.0±0.1	55.5±0.1	58.8±0.1	62.4±0.1	65.6±0.1	68.7±0.1
	Naive Bayes (Tong et al., 2026)	51.3±0.2	52.7±0.1	54.5±0.1	56.7±0.4	57.5±0.4	58.7±0.3
	Merge-Order with Bernoulli (ours)	**62.0±0.1**	**73.7±0.1**	**79.4±0.1**	**83.7±0.0**	**86.9±0.1**	**89.2±0.0**
	Merge-Order with Bradley-Terry (ours)	58.2±0.0	68.1±0.1	74.0±0.0	79.1±0.0	83.0±0.1	86.2±0.1

Table 6: **Leakage is still present at smaller vocabulary sizes.** TPR@1%FPR against undefended tokenizers. Bold marks the strongest MIA in each column. The last row gives BPT.

Access model	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$				
		10,000	15,000	20,000	30,000	40,000
Token-Count	Token-Count Attack (ours)	1.1±0.1	1.5±0.1	2.4±0.2	4.1±0.2	6.4±0.1
Token-Identity	Token-Identity Attack (ours)	**4.8±0.0**	5.1±0.1	5.8±0.1	7.6±0.1	9.6±0.1
Merge-Order	Merge-Order with Bernoulli (ours)	**4.8±0.1**	**5.8±0.1**	**7.4±0.1**	**11.0±0.1**	**14.4±0.1**
	Merge-Order with Bradley-Terry (ours)	3.9±0.0	4.6±0.0	5.6±0.0	8.6±0.0	11.6±0.1
BPT		3.25±0.02	3.96±0.01	4.24±0.00	4.51±0.00	4.63±0.00

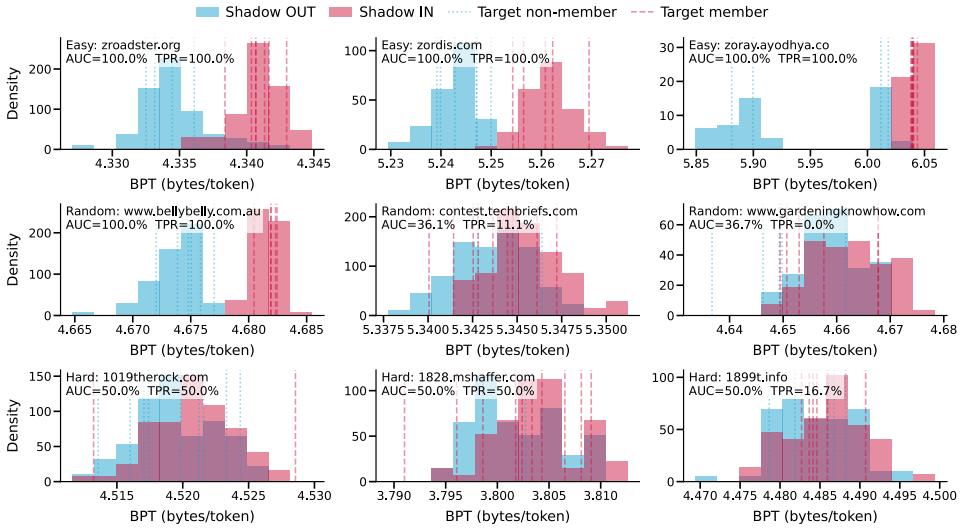

Figure 6: **Websites differ both in the separation and in the location of their IN and OUT distributions.** Raw BPT distributions at a vocabulary of 80,000 for nine candidate websites, grouped as in Figure 3 and each panel scaled to its own range.

Table 7: **Leakage is still present at smaller vocabulary sizes by ROC-AUC as well.** ROC-AUC (%) against undefended tokenizers at small vocabulary sizes. Bold marks the strongest MIA in each column. ROC-AUC counterpart of Table 6.

Access model	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$				
		10,000	15,000	20,000	30,000	40,000
Token-Count	Token-Count Attack (ours)	50.6±0.8	51.8±0.6	54.2±0.4	**58.9±0.3**	**63.6±0.2**
Token-Identity	Token-Identity Attack (ours)	**51.9±0.7**	**54.5±0.4**	**57.0±0.3**	58.7±0.3	60.5±0.3
Merge-Order	Merge-Order with Bernoulli (ours)	51.7±0.0	52.9±0.0	54.5±0.1	58.3±0.1	62.0±0.1
	Merge-Order with Bradley-Terry (ours)	**51.9±0.0**	52.5±0.0	53.4±0.0	55.7±0.0	58.2±0.0

G.3 Mismatched Vocabulary Size

Here we vary whether the shadow tokenizers share the target's *vocabulary size*, which an adversary with access to someone else's tokenizer generally does not control. We fix them at a vocabulary of 110,000 against the undefended target and sweep the target vocabulary over the remaining five sizes, so that shadow and target differ in vocabulary size but in nothing else.

Table 8: **Vocabulary-size mismatch degrades but does not eliminate our attacks.** TPR@1%FPR against undefended targets, with shadow tokenizers fixed at a vocabulary of 110,000 and the target vocabulary size swept. The 110,000 column is the matched case from Table 1. Bold marks the strongest attack in each column within its access model.

Access model	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
		40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	Compression Rate (Tong et al., 2026)	**1.2±0.2**	1.3±0.2	1.2±0.0	1.3±0.1	1.3±0.1	1.3±0.1
	Token-Count Attack (ours)	0.9±0.1	**2.8±0.1**	**29.2±0.2**	**5.9±0.2**	**4.6±0.2**	**3.9±0.2**
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	**4.9±0.1**	10.2±0.2	13.1±0.3	1.9±0.2	1.5±0.1	1.3±0.1
	Token-Identity Attack (ours)	3.6±0.1	**11.2±0.2**	**27.0±0.1**	**5.8±0.2**	**4.5±0.3**	**3.4±0.3**
Merge-Order	Frequency (Tong et al., 2026)	2.0±0.2	1.9±0.2	8.9±0.1	1.4±0.1	1.1±0.1	1.3±0.1
	Naive Bayes (Tong et al., 2026)	0.0±0.0	0.0±0.0	3.6±0.1	0.0±0.0	0.0±0.0	0.0±0.0
	Merge-Order with Bernoulli (ours)	**13.0±0.4**	**23.4±0.9**	**35.7±0.2**	17.9±1.0	12.1±0.4	10.3±0.5
	Merge-Order with Bradley-Terry (ours)	11.7±0.2	21.0±0.3	29.0±0.3	**23.5±0.2**	**17.4±0.3**	**15.2±0.1**

Mismatch costs the adversary accuracy but does not disarm our MIAs, whereas the prior baselines collapse, with Naive Bayes reaching 0.0% at every mismatched size. Table 8 gives the full results, in which performance peaks at the matched vocabulary size and degrades in both directions. The Token-Count and Token-Identity Attacks degrade the most among our MIAs. A shadow tokenizer that stops its merge list at a different point induces IN and OUT distributions over a token universe that only partially overlaps the target's, to which the Merge-Order variants are more robust because a pairwise ordering constraint remains meaningful over the tokens the two vocabularies share, whereas the Token-Identity Attack scores exactly the token presences the mismatch perturbs most. The two Merge-Order variants and the Token-Identity Attack stay above the 1% FPR at every target size, so an adversary who does not know the defender's vocabulary size still infers membership better than random guessing. The Token-Count Attack is the exception, falling to 0.9% at a vocabulary of 40,000.

G.4 Estimating the Vocabulary Size by Query

Our MIAs train their shadow tokenizers at the vocabulary size of the target, which an adversary with only Token-Count access does not observe, and a mismatched size weakens every one of them. Here we test whether such an adversary can recover the vocabulary size from token-count queries alone and so attack in the matched setting.

Setup. A string s is a token of $T^\star$ exactly when $T^\star$ encodes it as one token, so one query answers

$$O(s) \;=\; \mathbf{1}[\mathcal{O}_{\text{count}}(T^\star)(s) = 1]\,.$$

The adversary queries the tokens of K of its shadow tokenizers in the order they were learned. With $s_k(r)$ the token that shadow tokenizer k learned at position r, the fraction of them the target contains,

$$p(r) \;=\; \frac{1}{K}\sum_{k=1}^{K} O\big(s_k(r)\big),$$

is close to 1 for $r < |V(T^\star)|$ and close to 0 beyond it, since tokenizers trained on the same distribution learn similar tokens in a similar order. The vocabulary size is therefore the area under p,

$$|V(T^\star)| \;\approx\; \sum_{r<V_{\max}} p(r) \;\approx\; a + \int_a^b p(r)\,dr,$$

where the second step counts each position below a window $[a, b]$ around the drop as 1 and each position above it as 0. Near the drop the orders of two tokenizers disagree, but a token the target lacks

Algorithm 8 Vocabulary-size estimation with token-count queries

Require: query access $\mathcal{O}_{\mathrm{count}}(T^\star)$, shadow tokenizers $T_1, \ldots, T_K$ trained to $V_{\max}$ tokens, stopping width g, window half-width h, stride Δ

$\ell \leftarrow 0$, $u \leftarrow V_{\max}$ {binary search for the drop of p}
while $u - \ell > g$ **do**
  $m \leftarrow \lfloor (\ell + u)/2 \rfloor$
  if $p(m) \geq 1/2$ **then** $\ell \leftarrow m$ **else** $u \leftarrow m$ {K queries}
end while
$r_c \leftarrow \lfloor (\ell + u)/2 \rfloor$, $a \leftarrow \max\{0, r_c - h\}$, $b \leftarrow \min\{V_{\max}, r_c + h\}$
query $p(r_j)$ at $r_j = a,\ a + \Delta, \ldots, b$ {K queries each}
$\widehat{V} \leftarrow a + \int_a^b p(r)\, dr$, by the trapezoidal rule over the r_j
return $\widehat{V}$

Table 9: **Token-count queries estimate the vocabulary size with low error.** Estimated vocabulary size (mean±standard deviation) and mean relative error over three undefended targets per size.

	$\lvert\mathcal{V}_{\mathrm{target}}\rvert$					
	40,000	80,000	110,000	140,000	170,000	200,000
Estimate	39,510±425	80,547±1,025	108,814±109	137,443±5,619	171,027±4,566	187,618±6,705
Error (%)	1.23	1.10	1.08	3.69	2.27	6.19

below its size and one it has above cancel in the integral. Algorithm 8 finds the window by a binary search on p and evaluates the integral by the trapezoidal rule. We use $K = 8$ shadow tokenizers trained to $V_{\max} = 200{,}000$, $g = 2{,}000$, $h = 6{,}000$ and $\Delta = 200$, which costs at most 544 queries per target, 56 for the search and 488 for the window, and estimate the size of three undefended targets at each of the six vocabulary sizes.

Results. Up to $170{,}000$, the mean relative error in Table 9 stays below 4% and no single estimate is off by more than 4.4%. At $200{,}000$ the estimator underestimates the size, with estimates between $180{,}168$ and $193{,}168$, since the shadow tokenizers stop at $V_{\max}$ and leave no positions above the drop to cancel the tokens the target lacks below it.

G.5 Mismatched Pages per Website

Setup. We vary whether the shadow tokenizers are trained on the target's number of pages per website, which an adversary generally does not know. Besides the 50 pages per website used throughout, we build nested corpora with 100 and 200 pages over the same $21{,}090$ websites, and train targets on P_T and shadow tokenizers on P_S pages per website, $P_T, P_S \in \{50, 100, 200\}$, at a vocabulary of $200{,}000$ with all other settings of the main experiments. Since we assume that the adversary does not know how many pages per website were used to train the target tokenizer, it evaluates each candidate website using P_S pages.

Results. Each of our MIAs is strongest when $P_S = P_T$ (Table 10), and in that matched case fewer pages per website make it stronger, so our MIAs do not need a website to contribute much text. A mismatch costs less when the shadow tokenizers use more pages than the target than when they use fewer, and the Token-Count Attack is the only one of our MIAs that stays above 20% in every setting.

G.6 Reduced Shadow Budget

Setup. Training shadow tokenizers is the adversary's main cost, and our MIAs spend 128 of them. Here we vary that budget over $N \in \{4, 8, 16, 32, 64, 128\}$ against the undefended target at a vocabulary of $110{,}000$, retraining the shadow tokenizers and re-running each attack from scratch at every N rather than subsampling a single larger run.

Results. The shadow budget separates our MIAs from the prior baselines in Figure 7. The baselines that estimate no per-token reference model are flat from as few as four to eight shadows onward, with Frequency between 8.8% and 9.2% at every budget, and Compression Rate, which uses no

Table 10: **Each of our MIAs is strongest when the shadow tokenizers are trained on the target's number of pages per website.** TPR@1%FPR at a vocabulary of 200,000 for targets trained on P_T and shadow tokenizers on P_S pages per website. Bold marks the best P_S for each MIA and target.

Access model	Attack	P_S	$P_T = 50$			$P_T = 100$			$P_T = 200$		
			50	100	200	50	100	200	50	100	200
Token-Count	Token-Count Attack (ours)		**60.9±0.3**	38.3±0.4	27.0±0.4	31.4±0.2	**49.2±0.4**	34.1±0.4	20.7±0.3	30.9±0.2	**41.1±0.3**
Token-Identity	Token-Identity Attack (ours)		**52.3±0.2**	8.2±0.1	8.9±0.1	6.3±0.1	**43.9±0.1**	9.7±0.1	5.9±0.1	7.7±0.1	**39.4±0.2**
Merge-Order	Merge-Order with Bernoulli (ours)		**56.2±0.2**	30.7±0.3	22.6±0.2	10.5±0.3	**50.6±0.2**	32.6±0.4	5.8±0.4	15.8±1.0	**45.7±0.3**
	Merge-Order with Bradley-Terry (ours)		**50.2±0.2**	29.4±0.2	22.4±0.2	2.5±0.1	**41.9±0.2**	29.5±0.2	1.7±0.1	5.1±1.5	**37.7±0.2**

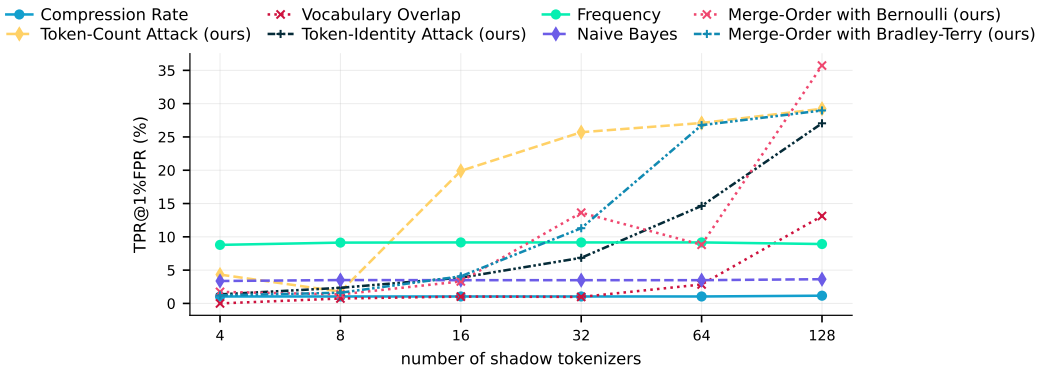

Figure 7: **Our attacks are strongest at the largest number of shadow tokenizers.** TPR@1%FPR against the undefended target at a vocabulary of 110,000 as a function of the number of shadow tokenizers, one line per attack.

shadows at all, staying close to random guessing throughout. Each of our MIAs instead reaches its highest value at $N = 128$, with the Token-Identity Attack rising from 1.4% at $N = 4$ to 27.0% and Merge-Order with Bradley-Terry from 1.2% to 29.0%. We therefore expect the 128 shadow tokenizers used elsewhere in the paper to understate what a better-resourced adversary would achieve.

G.7 Offline Setting

Setup. We run the offline variants of our MIAs against the undefended targets of the main experiments at all six vocabulary sizes, with the same candidate websites (Appendix C.4). As in the offline experiments of Carlini et al. (2022), we reuse the 128 shadow tokenizers of the main experiments and score each candidate website against only those trained without it, about half of them. Each of these still shares about half of its training websites with the target, as in the main experiments.

Results. Without shadow tokenizers trained with the candidate website, each of our MIAs stays above the 1% FPR at every vocabulary size (Table 11), so tokenizers remain vulnerable in the offline setting. Bernoulli reaches 8.0value, more than the Token-Identity Attack's 500.15 to 0.53 points and Merge-Order with Bernoulli by 2.1 to 7.7. The Token-Count Attack loses the least and is the strongest of our MIAs offline at every size, keeping at least 95% of its online TPR@1%FPR and reaching 58.0% at 200,000. The Token-Identity Attack keeps a share of its online value that grows with the vocabulary and reaches 44.1% at 200,000. Merge-Order with Bernoulli loses the most, keeping at most 41% of its online value and reaching 17.4% at 200,000.

H Document-Level Membership

Our MIAs detect membership when each candidate contributes a single document. We use the same 21,090 websites as in the main experiments. For each target or shadow tokenizer, the same websites remain members, but each contributes one selected page instead of 50. Each target and shadow tokenizer is trained using the selected page from each member website, and the adversary evaluates each candidate website using its selected page. Website membership therefore coincides

Table 11: **Our MIAs remain effective in the offline setting.** TPR@1%FPR against undefended tokenizers with both IN and OUT shadow tokenizers and with OUT shadow tokenizers only.

Attack	Setting	$\lvert\mathcal{V}_{\text{target}}\rvert$					
		40,000	80,000	110,000	140,000	170,000	200,000
Token-Count Attack	Online	6.4±0.1	18.6±0.1	29.2±0.2	41.1±0.3	51.6±0.3	60.9±0.2
	Offline	6.5±0.1	18.0±0.2	28.5±0.2	39.6±0.3	49.9±0.3	58.0±0.3
Token-Identity Attack	Online	9.6±0.1	19.0±0.1	27.0±0.1	35.7±0.1	44.3±0.2	52.3±0.2
	Offline	4.6±0.1	10.5±0.2	17.1±0.2	25.6±0.3	34.6±0.3	44.1±0.7
Merge-Order with Bernoulli	Online	14.4±0.1	27.0±0.2	35.7±0.2	43.8±0.3	50.7±0.3	56.2±0.2
	Offline	5.9±0.1	8.6±0.3	10.9±0.4	13.2±0.6	15.4±0.7	17.4±0.8

Table 12: **Our MIAs detect document-level membership under every access model.** TPR@1%FPR against undefended tokenizers when target and shadow tokenizers are trained using one selected page per member website.

Access model	Attack	TPR@1%FPR
Token-Count	Token-Count Attack (ours)	32.9±0.1
Token-Identity	Token-Identity Attack (ours)	84.1±0.1
Merge-Order	Merge-Order with Bernoulli (ours)	71.2±0.1
	Merge-Order with Bradley-Terry (ours)	66.9±0.1

with membership of the selected document. Table 12 reports TPR@1%FPR of 32.9% for the Token-Count Attack and 84.1% for the Token-Identity Attack. Merge-Order with Bernoulli and Merge-Order with Bradley-Terry reach 71.2% and 66.9%, respectively.

I ADDITIONAL DEFENSE RESULTS

I.1 MIN-COUNT FILTERING

Table 13: **Min-Count reduces every attack but removes none, and every threshold costs utility.** TPR@1%FPR against targets defended by the Min-Count filtering of Tong et al. (2026) at a vocabulary of 110,000, where the adversary trains its shadow tokenizers at the same $n_{\min}$ as the target. The last row gives the tokenizer's BPT at each threshold, and the first column the undefended tokenizer both quantities are measured against. Bold marks the strongest attack in each column.

Access model	Attack	Undefended	Min-Count (Tong et al., 2026)			
			$n_{\min}$=32	$n_{\min}$=64	$n_{\min}$=128	$n_{\min}$=256
Token-Count	Compression Rate (Tong et al., 2026)	1.2±0.0	1.1±0.0	1.1±0.0	1.1±0.0	1.0±0.0
	Token-Count Attack (ours)	29.2±0.2	11.5±0.2	7.8±0.2	4.9±0.3	2.9±0.1
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	13.1±0.3	10.4±0.2	10.0±0.2	8.3±0.1	5.0±0.1
	Token-Identity Attack (ours)	27.0±0.1	23.1±0.1	20.5±0.2	12.3±0.1	6.9±0.1
Merge-Order	Frequency (Tong et al., 2026)	8.9±0.1	9.9±0.1	8.5±0.1	5.3±0.0	2.8±0.0
	Naive Bayes (Tong et al., 2026)	3.6±0.1	3.7±0.1	3.5±0.1	2.2±0.0	1.5±0.0
	Merge-Order with Bernoulli (ours)	**35.7±0.2**	**31.9±0.3**	**28.9±0.2**	**19.6±0.2**	**12.6±0.1**
	Merge-Order with Bradley-Terry (ours)	29.0±0.3	25.7±0.4	23.9±0.2	15.7±0.1	9.9±0.1
BPT		4.87±0.01	4.72±0.01	4.60±0.01	4.34±0.01	4.03±0.01

An adversary who knows $n_{\min}$. Min-Count reduces leakage as $n_{\min}$ grows but never removes it, and it does so unevenly across access models, the same pattern the main paper reports at a vocabulary of 200,000. Table 13 gives every attack at a vocabulary of 110,000, where the adversary trains its shadow tokenizers at the target's own $n_{\min}$. The Token-Count Attack is the most affected, falling from 29.2% TPR@1%FPR undefended to 2.9% at $n_{\min} = 256$, which is what a defense aimed at exactly the low-count merges that carry the compression signal would produce. The two Merge-Order

Table 14: **The Token-Count Attack is the one that depends on knowing $n_{\min}$.** TPR@1%FPR at a vocabulary of 110,000 when the adversary trains its shadow tokenizers without the defense against a target defended by the Min-Count filtering of Tong et al. (2026). The BPT row and the undefended column are those of Table 13, since the target tokenizers are the same and only the shadow tokenizers change. Bold marks the strongest attack in each column.

Access model	Attack	Undefended	Min-Count (Tong et al., 2026)			
			$n_{\min}$=32	$n_{\min}$=64	$n_{\min}$=128	$n_{\min}$=256
Token-Count	Compression Rate (Tong et al., 2026)	1.2±0.0	1.1±0.0	1.1±0.0	1.1±0.0	1.0±0.0
	Token-Count Attack (ours)	29.2±0.2	1.0±0.0	1.0±0.0	1.1±0.0	1.0±0.0
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	13.1±0.3	11.8±0.3	16.1±0.1	4.5±0.2	0.7±0.0
	Token-Identity Attack (ours)	27.0±0.1	22.5±0.2	20.4±0.2	6.7±0.1	2.5±0.0
Merge-Order	Frequency (Tong et al., 2026)	8.9±0.1	9.9±0.1	8.5±0.1	5.3±0.0	2.8±0.0
	Naive Bayes (Tong et al., 2026)	3.6±0.1	3.7±0.1	3.5±0.1	2.2±0.0	1.5±0.0
	Merge-Order with Bernoulli (ours)	**35.7±0.2**	**31.6±0.3**	**28.9±0.3**	**19.3±0.2**	**12.6±0.1**
	Merge-Order with Bradley-Terry (ours)	29.0±0.3	26.3±0.3	24.5±0.2	16.2±0.1	10.0±0.1
BPT		4.87±0.01	4.72±0.01	4.60±0.01	4.34±0.01	4.03±0.01

variants are the least affected, with Bernoulli still reaching 12.6% at $n_{\min} = 256$, more than four times the Token-Count Attack at the same setting, since removing rare tokens leaves the order the corpus imposed on the surviving merges intact.

An adversary who does not know $n_{\min}$. Whether $n_{\min}$ has to be kept secret for Min-Count to protect anything depends entirely on the access model. Table 14 keeps the Min-Count-defended target but trains the shadow tokenizers without the defense, leaving everything else unchanged from the setting above, so the adversary no longer knows the threshold it is attacking. The Token-Count Attack collapses to the 1% FPR at every threshold, falling from 11.5% TPR@1%FPR with matching shadow tokenizers to 1.0% without at $n_{\min} = 32$, so what protection Min-Count offers under *Token-Count* access rests on the adversary not knowing the threshold. Merge-Order with Bernoulli is unaffected, reaching 31.6% without the matching shadow tokenizers against 31.9% with them at $n_{\min} = 32$, so the ordering signal transfers regardless and Min-Count does not protect the merge list even when $n_{\min}$ is public.

Utility cost. The protection Min-Count provides comes at a utility cost that rises with the vocabulary size. At a vocabulary of 110,000, BPT falls from 4.87 undefended to 4.03 at $n_{\min} = 256$, a 17% relative drop, and at a vocabulary of 200,000 the same setting reaches 3.76, the value the utility row of Table 2 reports. Even at that cost the defense remains incomplete, since at $n_{\min} = 256$ and a vocabulary of 110,000 the strongest MIA still reaches 12.6% TPR@1%FPR, more than an order of magnitude above the 1% FPR. Min-Count therefore reduces the Token-Count signal at a real utility cost while leaving the Merge-Order signal unprotected, which is the failure mode DP-BPE is designed to address.

I.2 DP-BPE

Under DP-BPE, no MIA we evaluate does better than chance at any privacy budget, including the two that are strongest against the undefended tokenizer. Table 17 gives every attack across the budgets $\varepsilon \in \{1, 3, 10\}$ and all six vocabulary sizes, and none reaches 1.3% TPR@1%FPR anywhere in it. Since this already holds at $\varepsilon = 10$, the loosest budget we evaluate, tightening the budget adds no further protection and only reduces utility, which is why the choice among these budgets is governed by the BPT row of that table rather than by attack success.

I.3 Full Results Tables

Tables 15 to 18 give every attack, including the prior baselines, at every defense level and every vocabulary size, where Table 2 reports our MIAs at a vocabulary of 200,000 and at three of the four Min-Count thresholds, Table 19 carries every setting of Softmax BPE at that vocabulary size, and Table 20 carries $\lambda = 20$ at all six. Table 15 adds $n_{\min} = 128$ and the remaining vocabulary sizes.

Under Min-Count the leakage falls as $n_{\min}$ grows at every vocabulary size, but no MIA of ours is brought down to the 1% FPR, the weakest cell any of them reaches being 2.9% at $n_{\min} = 256$ and a

vocabulary of $110{,}000$. Under DP-BPE, by contrast, no attack reaches 1.3% TPR@1%FPR at any budget or vocabulary size, and ROC-AUC stays within 2.2 points of chance throughout, so the bolded cell in each of those columns marks sampling noise rather than a stronger MIA.

Table 15: **Min-Count reduces leakage at every vocabulary size without removing it.** TPR@1%FPR (%) for every attack against tokenizers defended by the Min-Count filtering of Tong et al. (2026), at every $n_{\min}$ and every vocabulary size, with shadow tokenizers trained at the target's own $n_{\min}$. Bold marks the strongest attack in each column within its access model and defense level. The last rows give the BPT each threshold costs. Expands the Min-Count columns of Table 2.

Access model	$n_{\min}$	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
			40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	32	Compression Rate (Tong et al., 2026)	1.1±0.1	1.1±0.1	1.1±0.0	1.1±0.0	1.1±0.1	1.1±0.1
		Token-Count Attack (ours)	**5.4±0.1**	**9.9±0.2**	**11.5±0.2**	**12.4±0.2**	**11.7±0.2**	**10.9±0.2**
	64	Compression Rate (Tong et al., 2026)	1.1±0.1	1.1±0.1	1.1±0.0	1.1±0.0	1.1±0.0	1.1±0.0
		Token-Count Attack (ours)	**4.8±0.2**	**7.8±0.3**	**7.8±0.2**	**6.9±0.3**	**6.5±0.3**	**6.0±0.2**
	128	Compression Rate (Tong et al., 2026)	1.1±0.1	1.1±0.0	1.1±0.0	1.0±0.0	1.1±0.0	1.0±0.0
		Token-Count Attack (ours)	**4.4±0.3**	**5.9±0.2**	**4.9±0.3**	**4.6±0.3**	**4.4±0.4**	**3.8±0.2**
	256	Compression Rate (Tong et al., 2026)	1.1±0.0	1.1±0.0	1.0±0.0	1.1±0.0	1.0±0.0	1.1±0.0
		Token-Count Attack (ours)	**4.0±0.2**	**3.6±0.2**	**2.9±0.1**	**3.2±0.1**	**3.2±0.1**	**3.0±0.1**
Token-Identity	32	Vocabulary Overlap (Tong et al., 2026)	4.0±0.1	9.2±0.1	10.4±0.2	12.0±0.3	13.0±0.2	13.6±0.3
		Token-Identity Attack (ours)	**6.8±0.1**	**15.7±0.1**	**23.1±0.1**	**30.2±0.2**	**32.3±0.1**	**31.3±0.1**
	64	Vocabulary Overlap (Tong et al., 2026)	3.8±0.1	9.0±0.1	10.0±0.2	10.2±0.2	11.0±0.3	11.5±0.3
		Token-Identity Attack (ours)	**6.6±0.1**	**15.2±0.1**	**20.5±0.2**	**19.0±0.2**	**18.5±0.2**	**18.1±0.2**
	128	Vocabulary Overlap (Tong et al., 2026)	3.6±0.1	8.1±0.1	8.3±0.1	8.5±0.1	8.7±0.1	8.6±0.1
		Token-Identity Attack (ours)	**6.4±0.1**	**13.4±0.1**	**12.3±0.1**	**11.7±0.1**	**11.3±0.1**	**11.1±0.1**
	256	Vocabulary Overlap (Tong et al., 2026)	3.7±0.0	4.9±0.0	5.0±0.1	5.0±0.1	5.1±0.1	5.1±0.1
		Token-Identity Attack (ours)	**6.3±0.1**	**7.4±0.1**	**6.9±0.1**	**6.7±0.1**	**6.6±0.1**	**6.4±0.1**
Merge-Order	32	Frequency (Tong et al., 2026)	2.5±0.1	6.6±0.1	9.9±0.1	14.2±0.1	15.4±0.1	15.1±0.1
		Naive Bayes (Tong et al., 2026)	1.2±0.0	2.3±0.1	3.7±0.1	6.6±0.3	6.9±0.5	7.0±0.4
		Merge-Order with Bernoulli (ours)	**11.6±0.2**	**23.8±0.3**	**31.9±0.3**	**38.6±0.3**	**39.9±0.3**	**38.8±0.3**
		Merge-Order with Bradley-Terry (ours)	9.2±0.1	19.2±0.2	25.7±0.4	32.7±0.4	34.5±0.3	33.8±0.3
	64	Frequency (Tong et al., 2026)	2.5±0.1	6.4±0.0	8.5±0.1	8.0±0.1	8.0±0.1	8.0±0.1
		Naive Bayes (Tong et al., 2026)	1.2±0.0	2.3±0.1	3.5±0.1	3.6±0.1	3.6±0.1	3.6±0.1
		Merge-Order with Bernoulli (ours)	**11.5±0.2**	**23.3±0.3**	**28.9±0.2**	**27.3±0.2**	**26.7±0.3**	**26.2±0.3**
		Merge-Order with Bradley-Terry (ours)	9.0±0.1	18.8±0.2	23.9±0.2	22.9±0.3	22.5±0.2	22.1±0.3
	128	Frequency (Tong et al., 2026)	2.5±0.1	5.6±0.0	5.3±0.0	5.3±0.0	5.2±0.0	5.2±0.0
		Naive Bayes (Tong et al., 2026)	1.2±0.0	2.1±0.1	2.2±0.0	2.2±0.0	2.2±0.0	2.2±0.1
		Merge-Order with Bernoulli (ours)	**11.2±0.1**	**20.7±0.2**	**19.6±0.2**	**18.5±0.2**	**18.1±0.2**	**17.8±0.2**
		Merge-Order with Bradley-Terry (ours)	8.8±0.0	16.8±0.1	15.7±0.1	15.2±0.1	14.9±0.2	14.5±0.1
	256	Frequency (Tong et al., 2026)	2.5±0.1	2.9±0.0	2.8±0.0	2.8±0.0	2.8±0.1	2.8±0.1
		Naive Bayes (Tong et al., 2026)	1.2±0.0	1.5±0.0	1.5±0.0	1.5±0.0	1.5±0.0	1.5±0.0
		Merge-Order with Bernoulli (ours)	**11.0±0.1**	**13.4±0.2**	**12.6±0.1**	**12.1±0.2**	**11.8±0.1**	**11.4±0.2**
		Merge-Order with Bradley-Terry (ours)	8.5±0.1	10.6±0.1	9.9±0.1	9.6±0.1	9.3±0.1	9.0±0.1
BPT	32		4.57±0.01	4.71±0.01	4.72±0.01	4.71±0.01	4.68±0.01	4.66±0.01
	64		4.54±0.01	4.63±0.01	4.60±0.01	4.52±0.01	4.47±0.01	4.42±0.01
	128		4.48±0.01	4.46±0.01	4.34±0.01	4.26±0.01	4.19±0.01	4.12±0.01
	256		4.39±0.01	4.18±0.01	4.03±0.01	3.92±0.01	3.83±0.01	3.76±0.01

Table 16: **The Min-Count trend is the same by ROC-AUC.** ROC-AUC (%) for every attack against tokenizers defended by the Min-Count filtering of Tong et al. (2026), at every $n_{\min}$ and every vocabulary size, with shadow tokenizers trained at the target's own $n_{\min}$. Bold marks the strongest attack in each column within its access model and defense level.

Access model	$n_{\min}$	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
			40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	32	Compression Rate (Tong et al., 2026)	50.3±0.1	50.5±0.1	50.7±0.1	50.8±0.1	50.8±0.1	50.8±0.1
		Token-Count Attack (ours)	**60.0±0.2**	**65.3±0.2**	**67.6±0.2**	**68.7±0.2**	**68.1±0.3**	**67.0±0.2**
	64	Compression Rate (Tong et al., 2026)	50.3±0.1	50.5±0.1	50.6±0.1	50.6±0.1	50.6±0.1	50.6±0.2
		Token-Count Attack (ours)	**59.1±0.3**	**62.6±0.1**	**62.9±0.3**	**61.3±0.2**	**60.8±0.2**	**60.6±0.4**
	128	Compression Rate (Tong et al., 2026)	50.3±0.1	50.5±0.1	50.5±0.1	50.5±0.1	50.5±0.1	50.5±0.1
		Token-Count Attack (ours)	**57.8±0.2**	**59.3±0.2**	**58.0±0.4**	**57.3±0.4**	**56.6±0.3**	**56.3±0.6**
	256	Compression Rate (Tong et al., 2026)	50.3±0.1	50.4±0.1	50.4±0.1	50.4±0.1	50.3±0.1	50.3±0.1
		Token-Count Attack (ours)	**57.2±0.4**	**55.2±0.4**	**54.4±0.3**	**54.7±0.4**	**54.0±0.4**	**53.8±0.2**
Token-Identity	32	Vocabulary Overlap (Tong et al., 2026)	53.7±0.1	58.3±0.1	60.8±0.3	63.6±0.2	64.5±0.2	64.5±0.2
		Token-Identity Attack (ours)	**60.0±0.3**	**65.4±0.1**	**69.5±0.1**	**72.9±0.1**	**73.6±0.2**	**72.8±0.1**
	64	Vocabulary Overlap (Tong et al., 2026)	53.3±0.3	58.3±0.1	60.7±0.1	60.3±0.2	60.3±0.1	60.3±0.1
		Token-Identity Attack (ours)	**59.8±0.4**	**65.1±0.1**	**68.3±0.2**	**66.8±0.3**	**65.8±0.4**	**65.9±0.3**
	128	Vocabulary Overlap (Tong et al., 2026)	53.3±0.2	57.6±0.1	57.2±0.2	57.0±0.2	56.8±0.2	56.9±0.1
		Token-Identity Attack (ours)	**59.4±0.5**	**64.3±0.2**	**63.1±0.3**	**62.1±0.5**	**62.1±0.3**	**61.8±0.3**
	256	Vocabulary Overlap (Tong et al., 2026)	53.0±0.3	54.4±0.1	54.1±0.1	54.0±0.2	54.1±0.1	53.9±0.1
		Token-Identity Attack (ours)	**59.8±0.5**	**59.8±0.4**	**59.8±0.3**	**59.1±0.5**	**58.0±0.4**	**59.1±0.6**
Merge-Order	32	Frequency (Tong et al., 2026)	52.0±0.1	55.4±0.1	58.7±0.1	61.9±0.1	62.7±0.1	62.4±0.1
		Naive Bayes (Tong et al., 2026)	50.7±0.1	52.7±0.1	54.5±0.1	56.9±0.3	57.3±0.4	57.4±0.4
		Merge-Order with Bernoulli (ours)	**61.2±0.0**	**72.6±0.1**	**78.0±0.1**	**81.8±0.1**	**82.4±0.1**	**81.7±0.1**
		Merge-Order with Bradley-Terry (ours)	57.0±0.0	66.7±0.1	72.1±0.1	76.5±0.1	77.3±0.1	76.8±0.1
	64	Frequency (Tong et al., 2026)	52.0±0.1	55.2±0.1	57.8±0.1	57.3±0.1	57.1±0.1	56.9±0.1
		Naive Bayes (Tong et al., 2026)	50.7±0.1	52.7±0.1	54.2±0.1	54.2±0.1	54.3±0.1	54.3±0.1
		Merge-Order with Bernoulli (ours)	**61.0±0.1**	**72.1±0.1**	**76.1±0.1**	**75.1±0.1**	**74.5±0.1**	**73.9±0.1**
		Merge-Order with Bradley-Terry (ours)	56.9±0.0	66.2±0.0	70.3±0.1	69.4±0.1	69.0±0.1	68.5±0.1
	128	Frequency (Tong et al., 2026)	51.9±0.1	54.6±0.1	54.3±0.1	54.2±0.1	54.1±0.1	54.1±0.1
		Naive Bayes (Tong et al., 2026)	50.7±0.1	52.5±0.2	52.5±0.2	52.5±0.2	52.5±0.2	52.5±0.2
		Merge-Order with Bernoulli (ours)	**60.7±0.1**	**69.8±0.1**	**68.5±0.1**	**67.7±0.1**	**67.1±0.1**	**66.6±0.1**
		Merge-Order with Bradley-Terry (ours)	56.7±0.0	64.1±0.0	63.2±0.1	62.5±0.0	62.1±0.1	61.8±0.1
	256	Frequency (Tong et al., 2026)	51.8±0.1	52.4±0.1	52.3±0.1	52.3±0.1	52.2±0.1	52.2±0.1
		Naive Bayes (Tong et al., 2026)	50.7±0.1	51.1±0.1	51.2±0.1	51.2±0.1	51.2±0.1	51.2±0.1
		Merge-Order with Bernoulli (ours)	**60.4±0.1**	**62.7±0.1**	**61.6±0.1**	**61.1±0.0**	**60.5±0.0**	**60.2±0.1**
		Merge-Order with Bradley-Terry (ours)	56.5±0.0	58.3±0.0	57.7±0.0	57.4±0.0	57.1±0.0	56.8±0.0

Table 17: **Under DP-BPE no attack we evaluate does better than chance at any budget or vocabulary size.** TPR@1%FPR (%) for every attack against DP-BPE-defended tokenizers, at every ε and every vocabulary size, with shadow tokenizers trained at the target's own privacy budget. Bold marks the strongest attack in each column within its access model and budget, and the last rows give the BPT each budget costs.

Access model	ε	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
			40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	1	Compression Rate (Tong et al., 2026)	1.1±0.0	1.0±0.0	**1.1±0.0**	**1.1±0.0**	1.0±0.0	1.0±0.0
		Token-Count Attack (ours)	**1.1±0.0**	**1.0±0.0**	1.1±0.0	1.0±0.0	**1.0±0.1**	**1.1±0.1**
	3	Compression Rate (Tong et al., 2026)	**1.0±0.0**	**1.0±0.0**	**1.0±0.0**	**1.1±0.0**	1.0±0.0	**1.0±0.0**
		Token-Count Attack (ours)	1.0±0.0	0.9±0.0	1.0±0.0	1.0±0.0	**1.1±0.1**	1.0±0.0
	10	Compression Rate (Tong et al., 2026)	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0
		Token-Count Attack (ours)	**1.2±0.1**	**1.2±0.1**	**1.2±0.1**	**1.2±0.1**	**1.2±0.1**	**1.2±0.1**
Token-Identity	1	Vocabulary Overlap (Tong et al., 2026)	**1.0±0.1**	**1.0±0.0**	**1.1±0.0**	0.9±0.0	1.0±0.1	**1.0±0.0**
		Token-Identity Attack (ours)	1.0±0.0	1.0±0.0	1.1±0.0	**1.0±0.0**	**1.1±0.1**	0.9±0.0
	3	Vocabulary Overlap (Tong et al., 2026)	**1.1±0.0**	**1.1±0.1**	0.9±0.0	1.0±0.0	**1.1±0.0**	1.1±0.1
		Token-Identity Attack (ours)	1.0±0.0	0.9±0.0	**1.1±0.0**	**1.0±0.1**	1.0±0.1	**1.1±0.1**
	10	Vocabulary Overlap (Tong et al., 2026)	1.0±0.0	1.0±0.0	**1.0±0.0**	1.0±0.0	**1.0±0.0**	1.0±0.0
		Token-Identity Attack (ours)	**1.0±0.1**	**1.1±0.0**	1.0±0.0	**1.0±0.1**	1.0±0.1	**1.1±0.0**
Merge-Order	1	Frequency (Tong et al., 2026)	1.0±0.0	1.0±0.0	1.0±0.0	1.1±0.0	1.1±0.1	1.0±0.0
		Naive Bayes (Tong et al., 2026)	**1.1±0.0**	**1.1±0.0**	**1.1±0.0**	**1.1±0.0**	1.1±0.0	**1.1±0.0**
		Merge-Order with Bernoulli (ours)	0.8±0.0	0.7±0.0	0.7±0.0	0.7±0.0	0.8±0.0	0.8±0.0
		Merge-Order with Bradley-Terry (ours)	1.0±0.1	0.9±0.1	1.1±0.1	1.0±0.0	**1.1±0.1**	1.0±0.1
	3	Frequency (Tong et al., 2026)	1.0±0.0	1.0±0.1	1.1±0.1	1.0±0.0	1.1±0.1	1.0±0.0
		Naive Bayes (Tong et al., 2026)	**1.1±0.0**	**1.1±0.1**	1.1±0.0	**1.1±0.1**	**1.1±0.0**	**1.1±0.0**
		Merge-Order with Bernoulli (ours)	0.6±0.0	0.6±0.0	0.6±0.0	0.6±0.0	0.6±0.0	0.7±0.0
		Merge-Order with Bradley-Terry (ours)	1.0±0.0	1.0±0.0	**1.1±0.1**	1.1±0.1	1.0±0.1	1.1±0.0
	10	Frequency (Tong et al., 2026)	1.0±0.0	1.0±0.0	1.1±0.0	1.0±0.0	1.0±0.0	1.0±0.0
		Naive Bayes (Tong et al., 2026)	**1.1±0.0**	**1.1±0.0**	**1.1±0.0**	**1.1±0.0**	**1.1±0.1**	**1.1±0.0**
		Merge-Order with Bernoulli (ours)	0.4±0.0	0.5±0.0	0.4±0.0	0.4±0.0	0.4±0.0	0.4±0.0
		Merge-Order with Bradley-Terry (ours)	1.0±0.1	1.0±0.0	1.0±0.1	1.0±0.1	1.1±0.0	1.0±0.0
BPT	1		3.72±0.01	3.72±0.01	3.72±0.01	3.72±0.01	3.73±0.01	3.73±0.01
	3		4.29±0.01	4.29±0.01	4.29±0.01	4.29±0.01	4.29±0.01	4.29±0.01
	10		4.72±0.01	4.72±0.01	4.72±0.01	4.72±0.01	4.72±0.01	4.72±0.01

Table 18: **ROC-AUC under DP-BPE stays at chance throughout.** ROC-AUC (%) for every attack against DP-BPE-defended tokenizers, at every ε and every vocabulary size, with shadow tokenizers trained at the target's own privacy budget. Bold marks the strongest attack in each column within its access model and budget.

Access model	ε	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
			40,000	80,000	110,000	140,000	170,000	200,000
Token-Count	1	Compression Rate (Tong et al., 2026)	**50.2±0.1**	**50.1±0.1**	**50.2±0.1**	**50.2±0.1**	**50.2±0.1**	**50.2±0.1**
		Token-Count Attack (ours)	50.1±0.1	50.0±0.2	50.0±0.1	50.1±0.1	50.1±0.1	49.9±0.1
	3	Compression Rate (Tong et al., 2026)	50.2±0.1	50.2±0.1	50.2±0.1	50.2±0.1	50.2±0.1	50.2±0.1
		Token-Count Attack (ours)	**50.6±0.2**	**50.2±0.1**	**50.3±0.1**	**50.2±0.1**	**50.6±0.2**	**50.3±0.1**
	10	Compression Rate (Tong et al., 2026)	50.2±0.1	50.2±0.1	50.2±0.1	50.2±0.1	50.2±0.1	50.2±0.1
		Token-Count Attack (ours)	**51.8±0.1**	**51.7±0.2**	**51.9±0.1**	**52.1±0.1**	**51.9±0.0**	**52.1±0.1**
Token-Identity	1	Vocabulary Overlap (Tong et al., 2026)	50.0±0.1	**50.3±0.1**	50.0±0.1	**50.2±0.1**	**50.1±0.1**	**50.0±0.1**
		Token-Identity Attack (ours)	**50.1±0.1**	50.0±0.2	**50.1±0.2**	50.2±0.1	50.1±0.1	49.9±0.1
	3	Vocabulary Overlap (Tong et al., 2026)	49.9±0.1	50.0±0.2	**50.1±0.1**	**50.0±0.1**	**50.1±0.1**	50.1±0.1
		Token-Identity Attack (ours)	**50.1±0.1**	**50.1±0.1**	49.8±0.1	50.0±0.2	50.0±0.1	**50.2±0.1**
	10	Vocabulary Overlap (Tong et al., 2026)	50.2±0.1	50.1±0.1	**50.2±0.1**	50.1±0.1	50.1±0.1	50.1±0.1
		Token-Identity Attack (ours)	**50.4±0.1**	**50.4±0.1**	50.0±0.1	**50.3±0.1**	**50.2±0.2**	**50.3±0.1**
Merge-Order	1	Frequency (Tong et al., 2026)	50.0±0.2	49.8±0.1	50.0±0.1	50.0±0.1	50.1±0.2	49.9±0.2
		Naive Bayes (Tong et al., 2026)	**50.1±0.2**	**50.1±0.2**	**50.1±0.2**	**50.1±0.2**	**50.1±0.2**	**50.1±0.2**
		Merge-Order with Bernoulli (ours)	50.1±0.0	50.0±0.0	50.0±0.0	50.0±0.0	49.9±0.0	50.0±0.0
		Merge-Order with Bradley-Terry (ours)	50.1±0.1	49.9±0.2	50.1±0.1	50.0±0.2	50.0±0.1	50.0±0.1
	3	Frequency (Tong et al., 2026)	**50.2±0.2**	**50.2±0.2**	**50.2±0.2**	**50.2±0.2**	50.1±0.2	50.1±0.1
		Naive Bayes (Tong et al., 2026)	50.2±0.2	50.1±0.2	50.2±0.2	50.1±0.2	**50.2±0.2**	**50.1±0.2**
		Merge-Order with Bernoulli (ours)	50.0±0.0	50.0±0.0	50.0±0.0	50.0±0.0	50.0±0.0	50.0±0.0
		Merge-Order with Bradley-Terry (ours)	50.2±0.1	50.0±0.1	50.1±0.2	49.9±0.2	49.9±0.1	49.9±0.1
	10	Frequency (Tong et al., 2026)	50.1±0.1	50.1±0.1	50.1±0.1	50.1±0.1	50.0±0.1	50.0±0.1
		Naive Bayes (Tong et al., 2026)	**50.2±0.2**	**50.2±0.2**	**50.2±0.2**	**50.2±0.2**	**50.2±0.2**	**50.2±0.2**
		Merge-Order with Bernoulli (ours)	50.0±0.0	50.0±0.0	50.0±0.0	50.0±0.0	50.0±0.0	50.0±0.0
		Merge-Order with Bradley-Terry (ours)	50.1±0.1	49.8±0.1	49.8±0.1	50.1±0.1	50.1±0.1	50.0±0.1

Table 19: **Softmax BPE either destroys utility or leaves the leakage in place.** TPR@1%FPR for every attack against tokenizers defended by the Softmax BPE of Tong et al. (2026) at a vocabulary of 200,000, with the undefended tokenizer as the reference and BPT in the last row. At $\lambda \in \{1, 10\}$ no attack reaches 7.1% and the tokenizer keeps at most 3.51 BPT against the 4.92 of the undefended one, while at $\lambda = 20$ the utility returns and so does the leakage our MIAs exploit. Bold marks the strongest attack in each column within its access model.

Access model	Attack	Undefended	Softmax BPE (Tong et al., 2026)		
			$\lambda = 1$	$\lambda = 10$	$\lambda = 20$
Token-Count	Compression Rate (Tong et al., 2026)	1.3±0.0	**1.1±0.0**	**1.1±0.1**	1.2±0.1
	Token-Count Attack (ours)	**60.9±0.2**	1.0±0.1	1.0±0.1	**58.6±0.3**
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	17.8±0.3	1.6±0.1	2.8±0.3	7.1±0.3
	Token-Identity Attack (ours)	**52.3±0.2**	**4.6±0.1**	**4.6±0.1**	**46.4±0.2**
Merge-Order	Frequency (Tong et al., 2026)	10.9±0.4	3.4±0.3	6.3±1.0	9.9±0.3
	Naive Bayes (Tong et al., 2026)	8.5±0.4	**5.5±0.2**	**7.0±0.7**	10.3±0.1
	Merge-Order with Bernoulli (ours)	**56.2±0.2**	4.3±0.1	4.5±0.1	**50.4±0.2**
	Merge-Order with Bradley-Terry (ours)	50.2±0.2	3.5±0.0	3.6±0.0	45.5±0.2
BPT		4.92±0.01	1.22±0.01	3.51±0.48	4.92±0.01

Table 20: **At $\lambda = 20$ Softmax BPE leaves the leakage in place at every vocabulary size.** TPR@1%FPR and ROC-AUC (%) for every attack against tokenizers defended by the Softmax BPE of Tong et al. (2026) at $\lambda = 20$, at every vocabulary size. Bold marks the strongest attack in each column within its access model. Expanded counterpart of the $\lambda = 20$ column of Table 19.

Access model	Attack	$\lvert\mathcal{V}_{\text{target}}\rvert$					
		40,000	80,000	110,000	140,000	170,000	200,000
TPR@1%FPR (%)							
Token-Count	Compression Rate (Tong et al., 2026)	1.1±0.1	1.1±0.1	1.2±0.1	1.2±0.1	1.2±0.1	1.2±0.1
	Token-Count Attack (ours)	**5.7±0.1**	**17.4±0.2**	**27.6±0.2**	**38.5±0.3**	**48.9±0.2**	**58.6±0.3**
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	4.3±0.2	4.7±0.3	5.2±0.3	6.0±0.3	6.7±0.3	7.1±0.3
	Token-Identity Attack (ours)	**8.3±0.1**	**17.0±0.1**	**23.7±0.2**	**31.5±0.2**	**39.3±0.3**	**46.4±0.2**
Merge-Order	Frequency (Tong et al., 2026)	2.2±0.1	5.5±0.1	7.0±0.1	9.2±0.2	9.7±0.3	9.9±0.3
	Naive Bayes (Tong et al., 2026)	4.0±0.1	2.3±0.1	4.1±0.1	7.4±0.0	9.5±0.1	10.3±0.1
	Merge-Order with Bernoulli (ours)	**9.9±0.1**	**20.8±0.2**	**29.3±0.2**	**37.5±0.1**	**44.3±0.1**	**50.4±0.2**
	Merge-Order with Bradley-Terry (ours)	8.2±0.1	17.5±0.1	25.0±0.2	32.7±0.1	39.5±0.3	45.5±0.2
ROC-AUC (%)							
Token-Count	Compression Rate (Tong et al., 2026)	50.5±0.2	50.8±0.2	51.1±0.2	51.4±0.2	51.5±0.2	51.7±0.2
	Token-Count Attack (ours)	**61.9±0.2**	**76.5±0.1**	**84.1±0.1**	**89.6±0.1**	**93.1±0.1**	**95.4±0.1**
Token-Identity	Vocabulary Overlap (Tong et al., 2026)	52.6±0.1	54.5±0.2	55.7±0.2	57.3±0.3	58.5±0.3	59.3±0.3
	Token-Identity Attack (ours)	**53.9±0.2**	**60.5±0.2**	**65.6±0.2**	**70.5±0.1**	**74.7±0.1**	**78.2±0.1**
Merge-Order	Frequency (Tong et al., 2026)	51.8±0.2	55.4±0.1	58.8±0.1	62.2±0.1	65.3±0.1	68.2±0.1
	Naive Bayes (Tong et al., 2026)	51.6±0.1	52.7±0.1	54.9±0.1	57.7±0.1	59.2±0.1	59.6±0.1
	Merge-Order with Bernoulli (ours)	**57.2±0.1**	**68.0±0.1**	**74.6±0.1**	**79.7±0.1**	**83.3±0.0**	**86.2±0.1**
	Merge-Order with Bradley-Terry (ours)	55.0±0.0	63.4±0.1	69.3±0.1	74.8±0.1	79.2±0.1	82.7±0.1