FALCON: A Universal Text-only Membership Inference Attack Framework against In-context Learning

Haitao Su, *Student Member, IEEE*, Yue Qin, Zhenhua Li, *Senior Member, IEEE*, Xin Miao, Yuan Zhou

***Abstract*—Membership inference attacks (MIAs) against in-context learning (ICL) serve as essential tools for privacy risk assessment and intellectual property safeguarding due to the use of small, private datasets for adaptation. However, most MIAs against language models require unrealistic, internal access or risk triggering built-in security mechanisms. In this paper, we propose FALCON (Flexible Attack on Language Context via ObfuscatioN), the first task-aware MIA framework against text-only model APIs. FALCON fully exploits the complexity of text obfuscation techniques and leverages the model's discrepancies in reconstructing obfuscated texts from seen versus unseen data as a strong membership signal, successfully bypassing application constraints and LLM safeguarding mechanisms.**

Through extensive experiments on six widely used LLMs, including four open-source models (Llama-2, Llama-3, Qwen-2.5, Ministral) and two commercial models (GPT-3.5, and GPT-4o-mini), five datasets from various domains for tasks including question answering, text classification and summarization, FALCON generally achieves over 95% attack success rates, significantly outperforming existing methods. An in-depth analysis of the impact of model scale shows that FALCON exploits a capacity-induced vulnerability, indicating that models with higher capabilities are more susceptible to our attack. Additionally, we explore three defense methods, highlighting role validation as a potential mechanism for safeguarding LLM privacy. We have open-sourced FALCON's modular, extensible codebase to support future research.

***Index Terms*—Membership inference attack, in-context learning, large language models, text obfuscation, model security.**

I. INTRODUCTION

In-context learning (ICL) has revolutionized LLM deployment, enabling models to handle complex prompts without fine-tuning or architectural changes [1]. By adding task-specific examples into the input context, service providers can rapidly customize their LLM-based applications for various domains and downstream applications with minimum annotated data [2]. This deployment approach has gained widespread adoption due to its minimal technical overhead and remarkable effectiveness, particularly in scenarios with restricted computational constraints or rapidly evolving task requirements, covering domains from legal consultation [3] to medical applications [4] and enterprise solutions [5].

Yue Qin is the corresponding author, with the School of Information, Central University of Finance and Economics, Beijing, China (email: qinyue@cufe.edu.cn).

Haitao Su is with the Qiuzhen College, Tsinghua University, Beijing, China (email: sht23@mails.tsinghua.edu.cn).

Zhenhua Li and Xin Miao are with the School of Software, Tsinghua University, Beijing, China (emails: lizhenhua1983@tsinghua.edu.cn, miaoxin@tsinghua.edu.cn).

Yuan Zhou is with the Yau Mathematical Sciences Center and the Department of Mathematical Sciences, Tsinghua University, and Beijing Institute of Mathematical Sciences and Applications, Beijing, China (email: yuanzhou@tsinghua.edu.cn).

While ICL offers substantial advantages, its integration into language models also enhances privacy risks, as adaptation often relies on small, private datasets, where task-specific prompts may inadvertently encode sensitive information. This vulnerability is particularly concerning in Membership Inference Attacks (MIAs), where adversaries analyze subtle patterns in model outputs to determine whether specific data samples were used during training or adaptation. Unlike traditional LLM deployments, ICL's context-driven learning introduces unique attack vectors, making it easier for adversaries to extract private details about individuals or proprietary datasets. This risk is especially critical in sensitive domains such as healthcare and legal services, where even indirect data leakage can compromise patient confidentiality, legal privilege, or intellectual property. Meanwhile, MIAs can also serve as data auditing tools, helping detect unauthorized use of private data in user-specific LLM adaptations [6]. A comprehensive understanding of MIAs is therefore essential for evaluating privacy risks and safeguarding intellectual property.

However, most existing MIAs against language models rely on probability-based methods, requiring access to loss values or token logits from the target model [7], [8], [9], [10], [11], [12], [13], [14], [15], [16]. Such a dependency on internal metrics is incompatible with many real-world LLM services. For instance, models deployed behind restrictive APIs, or embedded in applications like web interfaces and chatbotstypically only return generated text.

Recent research introduced text-only MIAs tailored for In-Context Learning (ICL), where attackers probe models by querying whether specific samples were encountered (i.e., INQUIRY attack) or by prompting with initial fragments of data to elicit memorized content (i.e., REPEAT attack) [17]. However, these methods risk triggering LLM security mechanisms, as such attack prompts may appear privacy-intrusive, leading to automated detection, flagged alerts, and ultimately refusal of response due to explicit attempts to extract sensitive information. Additionally, the BRAINWASH attack [17] explores a task-specific scenario where the LLM is limited to classification outputs rather than free-text responses. This attack is based on the assumption that LLMs resist manipulation when faced with unreasonable or incorrect queries on previously encountered correct answers. Yet, as LLMs advance in reasoning capabilities, such attacks are expected to lose effectiveness. These challenges highlight a critical gap in MIA-on-ICL research: the lack of stealthy , generalizable methods that operate with text-only access with no internal signals and avoid triggering security safeguards.

FALCON: design and implementation. To address this challenge, we propose FALCON (**F**lexible **A**ttack on **L**anguage

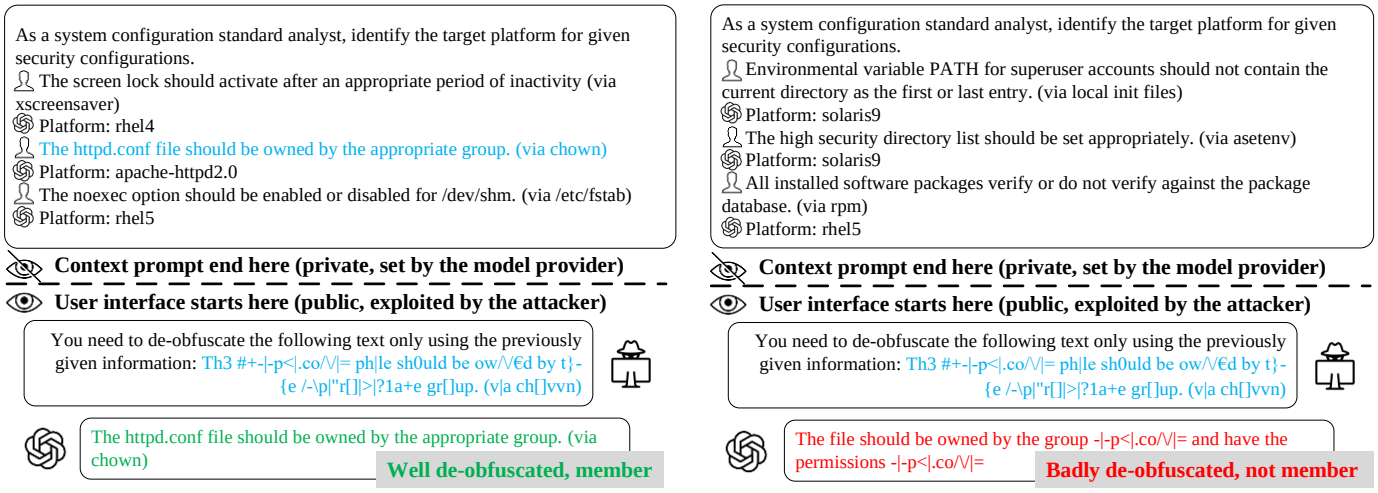

Fig. 1. An example of FALCON framework in the CCE [18] classification task. The attack prompt is simplified for clarity.

Context via **O**bfuscatio**N**), the first task-aware MIA against text-only model APIs. Unlike prior MIA methods primarily relying on external probing or manipulation techniques, FALCON redefines the attack paradigm by leveraging the model's intrinsic task-handling behavior as a membership signal. More specifically, FALCON reframes membership inference as a plausible operational need for preprocessing and input cleaning in real-world ML deployments. This design draws inspiration from the adversarial dynamics of text obfuscation, a widely observed tactic for evading content moderation systems (e.g., hate speech filters or illicit promotion detectors). For example, adversaries often replace characters in sensitive terms (e.g., writing "c@3h" for "cash") to bypass keyword-based detection, thereby necessitating a text-cleaning (i.e., de-obfuscation) step before reliable detection can occur. While de-obfuscation aims to reverse such modifications, it faces inherent challenges due to the high variability in obfuscation strategies (e.g., synonym substitutions, misspellings, leetspeak, and paraphrasing), the structural complexity of language, and ambiguity in modified texts. These challenges create a critical attack surface: **models exhibit discrepancies in reconstructing obfuscated versions of seen versus unseen data.** When processing samples obfuscated from their adaptation data, LLMs unconsciously leverage contextual priors and short-term memorization to "denoise" perturbations. Figure 1 illustrates an instance of how an LLM's ability differs in de-obfuscating text, based on whether the original text is present in the model's context prompt. By leveraging differential reconstruction patterns as membership signals, FALCON weaponizes the legitimate operational need for preprocessing in real-world ML deployments, thus increasing its versatility, effectiveness, and stealth.

We conduct an extensive evaluation of FALCON across six LLM models and five datasets spanning classification and generation, where it consistently surpasses text-only baselines and achieves attack success rates above 95% in most settings. Specifically, our method outperforms the second-best attack (e.g., REPEAT) by achieving 16.9% higher accuracy on average across six target LLMs for all datasets and tasks. Moreover, our analysis reveals a capacity-induced vulnerability: larger and more capable models are more susceptible to FALCON, producing a utility–privacy paradox that complicates defense design (see Section V-D for detailed analysis). Beyond accuracy, our attack demonstrates strong stealth under realistic detection. Across two detection techniques [19], [20], FALCON maintains very low explicit-refusal rates (0.6–6.5% depending on the model), whereas prior text-only MIAs are frequently filtered (as high as 37.4% for REPEAT and 23.6% for INQUIRY; see Table III, Section V-C). Furthermore, as a task-aware attack, FALCON can essentially bypass guardrail checks: even under role validation—our most vigorous defense, which produces the most significant reduction in ASR—considerable vulnerability persists, with attack success rates remaining at 88–99% across evaluated models and tasks (see Figure 13, Section VI).

Contributions. We summarize our contributions as follows:

- We introduce FALCON, the first task-aware membership inference attack for in-context learning systems with text-only query access. By reframing membership inference as a plausible preprocessing requirement, FALCON enables a stealthy, generalizable attack that remains effective against existing LLM security mechanisms.
- We demonstrate the effectiveness of FALCON through comprehensive experiments across diverse models, tasks, and datasets spanning multiple domains. To support future research in MIAs against language models, we release our code and implementation as a modular, extensible codebase[1], providing a robust and flexible foundation for further exploration.
- Experimental findings reveal a critical capacity-induced vulnerability of existing LLMs exposed to our attack. This property introduces a utility-privacy paradox to the mitigation, further intensifying the threat.
- We explore three potential defense methods against the proposed attack and empirically evaluate their effectiveness in mitigating membership inference risks.

[1]Our repository is available at https://github.com/Bug-001/Falcon-MIA/.

II. Background and Related Work

A. Large Language Models

Large Language Models (LLMs) have revolutionized natural language processing by leveraging massive text corpora to achieve exceptional proficiency in understanding and generating human-like text, enabling diverse real-world applications [21], [22]. These models have been successfully applied across a broad spectrum of fields and industries. For example, in data management systems, they are deployed to handle complex querying tasks and optimize system operations [23]. In software engineering, LLMs assist developers with code generation, bug detection, and documentation tasks [24]. Their applications also extend to specialized domains such as legal analysis [25], medical diagnosis [26], and various kinds of scientific research [27], where they demonstrate strong capabilities in domain-specific language understanding and task execution. The growing adoption of LLMs across diverse domains, particularly in professional and specialized fields, underscores the need for a comprehensive evaluation of their privacy risks.

In this study, our investigation spans multiple tasks and specialized fields to reflect the real-world complexity of LLM deployments and their associated privacy risks.

B. In-context Learning

In-context learning (ICL) has revolutionized task adaptation in natural language processing by allowing large language models to perform tasks through input demonstrations without modifying model parameters, enabling greater flexibility and efficiency [1], [2]. This approach works by first providing the model with task demonstrations as context before presenting a query, allowing it to generate responses by leveraging both the exemplars and its pre-trained knowledge [28], [29]. For instance, in classification tasks, a prompt may include labeled examples followed by an unlabeled instance, which the model then classifies based on the demonstrated patterns. By enabling task adaptation without parameter updates, ICL challenges traditional machine learning paradigms and proves especially valuable in rapidly evolving tasks or resource-constrained environments [30], [31].

Although theoretically ICL demonstrations and query inputs are fed into the model simultaneously, in many practical applications such as Retrieval-Augmented Generation [32] (RAG), service providers encapsulate the demonstrations as part of a predefined context. Users (or potential attackers) can only manipulate the query input and not the context prompt as shown in Figure 1.

C. Membership Inference Attack

Membership inference attacks (MIAs) pose a major privacy risk in machine learning, allowing adversaries to determine whether a specific data record was used in model training [33], [34], [35]. These attacks exploit differences in model behavior between training and unseen data, potentially leading to privacy breaches in sensitive applications. Formally, given a target model f_θ and a data point z, a membership inference attack can be modeled as a binary classification problem:

$$\mathcal{A}(z; f_\theta) \rightarrow \{0, 1\}, \tag{1}$$

where $\mathcal{A}$ represents the attack model, and the output indicates whether z was used in training f_θ (1) or not (0). These attacks typically rely on features such as prediction confidence, loss values, or output distributions [36].

Recent advancements in MIAs have improved effectiveness in black-box settings using methods like reference model calibration [37], loss trajectory analysis [38], and transferable attacks [39], while label-only scenarios exploit prediction consistency under perturbations to leak privacy [40], [41], demonstrating that even minimal model outputs risk exposing sensitive data.

Membership Inference against Language Models. By modeling the probability distribution of tokens, language models power many online services and are often trained on personal data like users' messages, search queries, and comments [42]. This reliance on sensitive data makes MIAs a critical privacy concern. At the same time, MIAs can be repurposed as auditing tools to detect unauthorized data usage and help enforce privacy regulations such as GDPR [43]. Earlier studies on MIAs for language models focused on pre-trained models by leveraging ranks of predicted tokens [43] or learned token embeddings [44]. More recently, researchers have highlighted the challenges of evaluating MIAs on LLMs [45] and proposed new attacks that target LLM pre-training data by exploiting signals like loss values or token logits [8], [9], [10], [11], [12], [13], [14], [15], [16]. These advances show that even the massive pre-training phase of LLMs is not immune to privacy leaks. To efficiently adapt LLMs to specific tasks, fine-tuning and in-context learning (ICL) have emerged as popular techniques. These adaptation stages use relatively small, often private datasets, making them especially susceptible to privacy leaks [46], [7], [17]. In the fine-tuning scenario, Fu et al. [7] introduced a probabilistic variation metric as a more reliable membership signal to determine if a given data record was used in training the target LLM. Similarly, Duan et al. presented a probability-based MIA against ICL [46]. Wen et al. [17] presented the first text-only MIA designed for ICL, the work most closely related to ours.

• *Text-only Membership Inference*. Among membership inference attacks on language models, text-only inference applies to the most restricted scenarios, where attackers can access only textual outputs without probability distributions [17]. In real-world settings, this poses a realistic privacy risk, as many chatbots embedded within applications provide only text-based outputs, making probability-based attacks infeasible. Adversaries, typically operating as end-users, lack privileged access to model internals and must rely solely on textual responses to infer membership information. This constraint significantly limits attackers' ability to exploit token-level confidence cues, making it more challenging to distinguish members from non-members based purely on discrete text outputs. Below, we introduce recent studies on text-only membership inference techniques.

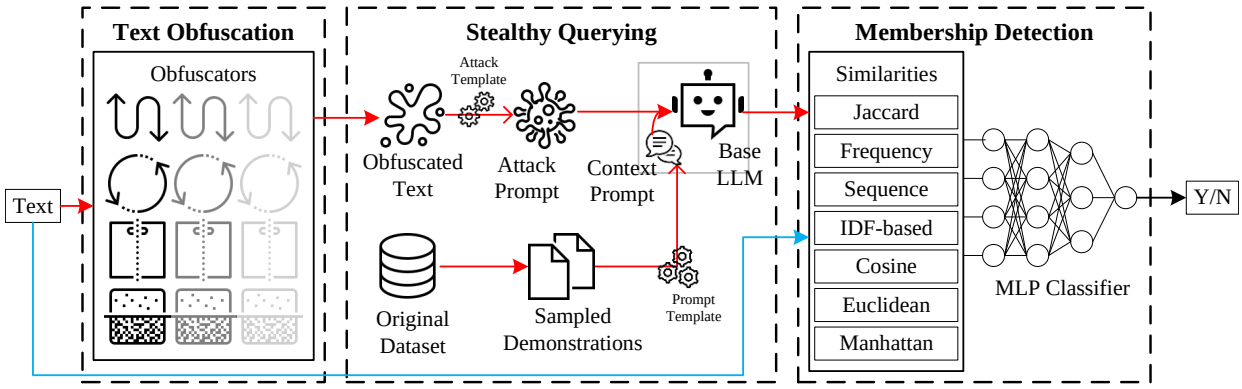

Fig. 2. The FALCON attack framework.

Given a target classifier f_θ^{tgt} and a text x with its ground truth label y, the GAP attack (introduced in [47] and adapted to LLM in [17]) as a naive baseline, using classification accuracy as a membership signal. The INQUIRY approach [17] directly queries f_θ^{tgt} about whether x has been seen before, considering positive responses as a membership signal. The REPEAT approach [17] leverages the model's ability to reconstruct complete text sequences from partial inputs. The BRAINWASH approach [17] exploits the model's resistance to misinformation by repeatedly querying it with incorrect labels until the model changes its prediction. The intuition is that models require more iterations to be convinced to produce incorrect outputs for familiar texts. The HYBRID attack [17] combines the semantic similarity signal from the REPEAT attack and the robustness signal from the BRAINWASH attack.

In this study, we extend the attack scenario to diverse LLM applications, including classification, question answering, and text summarization.

III. THREAT MODEL

We consider a text-only membership inference attack against in-context learning systems. The target exposed by the service provider, f_θ^{tgt}, is a large language model enhanced with a **context prompt vector** $\boldsymbol{c}$, formally defined as:

$$f_\theta^{\text{tgt}}(\cdot) = f_\theta^{\text{LLM}}((\boldsymbol{c}, \cdot)). \tag{2}$$

Here, f_θ^{LLM} is the base large language model with fixed parameters θ. The service constructs the context prompt vector $\boldsymbol{c}$ of in-context learning as follows:

$$\boldsymbol{c} = (p_T^{\text{sys}}, \Pi_T(d_1), \ldots \Pi_T(d_k)), \tag{3}$$

where p_T^{sys} is the system prompt which defines the task and role of the target model given task T, k is the number of demonstrations, and $\Pi_T(d_i)$ is the i-th demonstration which is generated from data point $d_i = (x_i, y_i)$ using the task-specific template Π_T. For example, if T is the question-answering task, a possible Π_T might be

$$\Pi_T(d_i) = (\text{“}\texttt{Answer the question:}x_i\text{”}, \\ \text{“}\texttt{This is the answer:}y_i\text{”}). \tag{4}$$

The attacker, on the user side, can send arbitrary text inputs u to the interface of f_θ^{tgt} and receive corresponding model responses $f_\theta^{\text{tgt}}(u)$. The attacker also has access to a public dataset $\mathcal{D}$ consisting of data points in the same domain as the ones used by the service provider. Unless stated otherwise, we assume the attacker knows the identity of the base LLM on which f_θ^{tgt} is built (e.g., model family/version), without requiring model weights. We show that this assumption can be relaxed through cross-model transferability in Section V-H. We assume the task-specific templates Π_T and the context prompt vector $\boldsymbol{c}$ are not available to the attacker, while the attacker has access to the public task description T (e.g., "summarize biomedical abstracts"), and can use this to construct a prompt template $\widehat{\Pi_T}$ and system prompt $\widehat{p_T^{sys}}$ that are semantically aligned with the task.

Given the setup above, the attacker aims to determine whether a query text s was used in constructing the context prompt vector $\boldsymbol{c}$. To this end, we define an attack function $\mathcal{A}(s; f_\theta^{\text{tgt}}) \to \{0, 1\}$ which outputs 1 when s is a substring of the concatenated string $x_i \| y_i$ for some $d_i = (x_i, y_i) \in d_1, d_2, \ldots, d_k$, and 0 otherwise.

IV. FALCON DESIGN

A. Overall Framework

FALCON is a text-only membership inference attack framework designed for in-context learning systems. Our key insight is that large language models exhibit stronger de-obfuscation capabilities for text sequences that have previously appeared in the context, compared to unseen sequences. Formally, given a query text s, we observe that the model can effectively recover s from its obfuscated forms when s was used to construct the context prompt vector, otherwise the model's recovery deviates significantly from s.

Based on this observation, as shown in Figure 2, we decompose the attack function $\mathcal{A}(s, f_\theta^{\text{tgt}})$ into three sequential steps:

1) **Text obfuscation:** A function $\mathcal{U}(s)$ that generates a vector of obfuscated variants $\boldsymbol{o} = (o_1, o_2, \ldots, o_m)$ from the input query text s.

2) **Stealthy querying:** A function $\mathcal{Q}(\boldsymbol{o}; f_\theta^{\text{tgt}}, T)$ that obtains the target model's responses through task-awarely crafted prompts, disguising the membership inference attempt as a benign de-obfuscation task. This yields a response vector

TABLE I
EXAMPLES OF DIFFERENT OBFUSCATION TECHNIQUES APPLIED TO THE TEXT "THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG" WITH DIFFERENT OBFUSCATION LEVELS (λ).

Technique	$\lambda = 0.3$	$\lambda = 0.9$
Character Swapping	The qucik borwn fox jmups over the lzay dog	The kqicu rwobn fxo jumps over the lyza dgo
Leet Speak	The quick |3rown fox jumps 0ver the la>_y <|og	-|-he qui<k |:ro\^/|/| ph0x _/um|*5 ov€r 7he l@>_y d()g
Unicode Lookalikes	Th̄é quick brown fox jumps over the lázy doğ	ẗȟe ϕcíck brȍwn fóx jūmps over the lázy dóğ
Synonym Substitution	The quick brown fox parachute all over the otiose dog	The fast Robert Brown play tricks jumps over the faineant wiener

$\boldsymbol{r} = (r_1, r_2, \ldots, r_m)$ where the entries correspond to each obfuscated variant in $\boldsymbol{o}$.

3) **Membership detection:** A function $\mathcal{M}(s, \boldsymbol{r})$ that analyzes the similarity between the original text and model responses to determine the membership, outputting 1 if s likely appears in $\boldsymbol{d}$ and 0 otherwise.

The complete attack can thus be expressed as:

$$\mathcal{A}_{\text{FALCON}}(s; f_\theta^{\text{tgt}}, T) = \mathcal{M}(s, \mathcal{Q}(\mathcal{U}(s); f_\theta^{\text{tgt}}, T)). \tag{5}$$

We elaborate on the three modules of FALCON in the following subsections.

B. Text Obfuscation

Our obfuscation framework supports multiple text transformation techniques, balancing semantic preservation with text distortion. The implemented methods include character-level transformations (character swapping and substitution via leet speak or unicode lookalikes) and word-level transformations (primarily synonym substitution). While designed to preserve human readability, these techniques effectively perturb the model's input token distribution.

1) Character Swapping: Character swapping randomly exchanges character positions within words. Formally, we denote character swapping as a function $u_{\text{swap}}(s, \lambda)$ that permutes characters in every word of s based on an obfuscation level $\lambda \in [0, 1]$. For each word, we perform $\lfloor n\lambda/2 + 0.5 \rfloor$ (the nearest integer to λ multiplied by half of the word length) random swaps without replacement, where the factor $1/2$ accounts for each swap affecting two positions.

2) Character Substitution: Character substitution systematically replaces characters with visually similar alternatives while maintaining readability. This includes two variants: (1) *leet speak* and (2) *unicode lookalikes*. Formally, we define character substitution as a function $u_{\text{sub}}(s, \lambda, \sigma)$, where $\sigma : \Sigma \to \Sigma'$ is a randomized character mapping that may produce different outputs for the same input character. Then the two variants of character substitution are $u_{\text{leet}}(s, \lambda) = u_{\text{sub}}(s, \lambda, \sigma_{\text{leet}})$ and $u_{\text{unicode}}(s, \lambda) = u_{\text{sub}}(s, \lambda, \sigma_{\text{unicode}})$. The examples are shown in Table I.

Content-aware Weight Scheme. In character substitution, we introduce a content-aware weighting scheme that assigns different obfuscation weights to words based on their semantic importance. Given a sentence-level weight λ, each word i receives a weight ω_i:

$$\omega_i = \begin{cases} \lambda & \text{if word } i \text{ is a } \textit{content word}, \\ \lambda/2 & \text{otherwise.} \end{cases} \tag{6}$$

Content words (nouns, verbs, adjectives) carry semantic meaning, while function words (articles, conjunctions, prepositions) maintain grammatical structure. This scheme prioritizes obfuscating semantic content while preserving structural hints for model de-obfuscation.

The total number of characters to be substituted is computed as $\lfloor \sum_i l_i \omega_i + 0.5 \rfloor$, where l_i is word length. Characters are then sampled with probabilities proportional to their word weights ω_i and transformed using the mapping σ.

3) Synonym Substitution: At the word level, we employ synonym substitution based on a key insight: while concepts can have multiple semantically equivalent expressions (e.g., "rapid" vs. "swift"), datasets typically favor specific word choices. This consistency makes synonym substitution effective for membership inference, as models tend to restore the exact wording they have encountered rather than semantically equivalent alternatives.

Formally, we define synonym substitution as a function $u_{\text{syn}}(s, \lambda) = u_{\text{syn}}(s, \lambda, \tau)$, where $\tau : V \to V$ is a word mapping function that may map a word to any of its synonyms within the vocabulary V. In our implementation, τ is constructed based on the WordNet synonym database [48]. For the input text $s = v_1 v_2 \ldots v_n$, each word v_i is transformed by τ with probability ω_i determined according to the content-aware weight scheme (Eq. (6)). The transformed text is the obfuscated text $u_{\text{syn}}(s, \lambda, \tau)$.

Table I demonstrates how different obfuscation techniques transform the same input text at various obfuscation levels and maintain the balance between information obscurity and readability, while targeting different aspects of text structure.

Finally, our complete obfuscation function $\mathcal{U}(s)$ generates a vector of obfuscated variants:

$$\mathcal{U}(s) = \boldsymbol{o} = (o_{u,\lambda} = u(s, \lambda))_{u \in U_{\text{base}}, \lambda \in \Lambda}, \tag{7}$$

where $U_{\text{base}} = \{u_{\text{swap}}, u_{\text{leet}}, u_{\text{unicode}}, u_{\text{syn}}\}$ is our set of base obfuscation functions and $\Lambda = \{0.6, 0.7, 0.8, 0.9, 1.0\}$ is the set of obfuscation levels. This comprehensive approach ensures that we capture different aspects of the model's de-obfuscation behavior across various transformation types and intensities.

C. Stealthy Querying

To elicit differential de-obfuscation behavior while maintaining stealth, we design the attack prompt $A_T(s)$ to remain superficially consistent with the intended ICL task T by explicitly assigning the model a role aligned with this task. Specifically, the prompt design embodies the following principles:

• *Role restriction:* The adversary positions the LLM in a role consistent with the original ICL task, while invoking a plausible preprocessing rationale (e.g., adversarial data cleaning or recovery from poisoning). This framing ensures that the queries remain superficially aligned with the demonstrated task. Moreover, it reduces reliance on the model's general language understanding, instead pressuring it to depend more heavily on pattern matching with its demonstrations.

• *Output control:* We provide step-by-step exemplars to constrain the output structure and format—first generating the de-obfuscated user query, then providing the corresponding answer. This ensures that the de-obfuscated text can be directly extracted and fed into the membership detector.

• *Cautious phrasing:* We carefully avoid suspicious instructions (e.g., "forget previous instructions") that might trigger the model's defense mechanisms, ensuring the task appears as a legitimate text processing exercise.

In all, the query prompt frames the attack as an innocuous role-playing and text processing task. Therefore, the query is more likely to bypass API filters or model defense mechanisms that specifically target privacy-intrusive queries. The complete prompt used in our attack is given in Appendix A in the supplementary material.

For each obfuscated variant o_i, a single query response r_i is obtained through:

$$r_i = f_\theta^{\text{tgt}}(A_T(o_i)), \tag{8}$$

where each response r_i is either a de-obfuscated text attempt or a claim that the model is unable to de-obfuscate the input. We treat failure/refusal responses as regular outputs in our detection pipeline by computing similarity scores against the original text (typically yielding low similarity), ensuring consistent handling across defended and undefended settings.

Finally, for $\boldsymbol{o} = (o_1, o_2, \ldots, o_m)$, the complete stealthy querying function $\mathcal{Q}(\boldsymbol{o}; f_\theta^{\text{tgt}}, T)$ generates a vector of responses $\boldsymbol{r}$:

$$\mathcal{Q}(\boldsymbol{o}; f_\theta^{\text{tgt}}, T) = \boldsymbol{r} = (r_1, r_2, \ldots, r_m). \tag{9}$$

D. Membership Detection

The core of FALCON's membership detection leverages the differential behavior in the target model's de-obfuscation capabilities between previously encountered and unseen sequences. To capture this signal comprehensively, we construct a feature matrix with a set of similarity measures between de-obfuscated responses $\boldsymbol{r}$ attained by Eq. (9) and the original query s, which are then processed by a neural network h_ϕ for final membership classification.

Now we first define the similarity measures, and then describe the design of our neural classifier h_ϕ.

Similarity Measures. We introduce the similarity measures of two texts (s_1 and s_2) that operate at both the lexical level and the semantic level. These two levels offer a complementary characterization of the target model's ability of de-obfuscation.

Lexical-level similarities. At the surface level, we examine both the basic text structure and word importance through the following five metrics:

• *Jaccard similarity* [49]: For any text s, we denote its word set as $W(s) = \{v : v \in s\}$, which extracts all unique words from the sequence. The Jaccard similarity measures the overlap between word sets, which is robust to word order variations:

$$\text{sim}_{\text{jaccard}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{|W(s_1) \cap W(s_2)|}{|W(s_1) \cup W(s_2)|}. \tag{10}$$

• *Frequency-based similarity* [50]: Let $\text{cnt}(s, v)$ denote the number of occurrences of the word v in the text s. The frequency-based similarity metric extends beyond simple set operations by considering the frequency distribution of words, which might be useful for longer texts where word repetition carries meaning:

$$\text{sim}_{\text{freq}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{\sum_v \min(\text{cnt}(s_1, v), \text{cnt}(s_2, v))}{\max(|s_1|, |s_2|)}. \tag{11}$$

• *Sequence similarity* [51]: Let $\text{LCS}(s_1, s_2)$ denote the longest common subsequence of words between s_1 and s_2. The sequence similarity metric is proportional to the length of the LCS, which is sensitive to the word order information, and can detect when the model preserves specific phrasal patterns:

$$\text{sim}_{\text{seq}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{|\text{LCS}(s_1, s_2)|}{\max(|s_1|, |s_2|)}. \tag{12}$$

• *IDF-weighted similarities*: According to [52], let

$$\text{IDF}(v) \stackrel{\text{def}}{=} \ln\left(\frac{|\mathcal{D}|}{1 + \sum_{(x,y)\in\mathcal{D}} \mathbb{1}(v \in x \vee v \in y)}\right) \tag{13}$$

denote the inverse document frequency of word v in the dataset $\mathcal{D}$, where $|\mathcal{D}|$ is the number of documents (i.e., the number of question-answer pairs)and $\mathbb{1}(\cdot)$ is the indicator function. The IDF trick enhances the basic Jaccard and frequency-based similarities by weighting each word according to its rarity:

$$\text{sim}_{\text{idf-j}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{\sum_{v\in W(s_1)\cap W(s_2)} \text{IDF}(v)}{\sum_{v\in W(s_1)\cup W(s_2)} \text{IDF}(v)}, \tag{14}$$

$$\text{sim}_{\text{idf-f}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{\sum_v \min(\text{cnt}(s_1, v), \text{cnt}(s_2, v)) \cdot \text{IDF}(v)}{\max(\sum_v \text{cnt}(s_1, v) \cdot \text{IDF}(v), \sum_v \text{cnt}(s_2, v) \cdot \text{IDF}(v))}. \tag{15}$$

By incorporating IDF weights, these metrics emphasize matches on rare, potentially more informative words while downplaying common function words, focusing more on the similarity of the "real content".

Semantic-level similarities. Complementary to the lexical-level measures, semantic-level similarity metrics can capture deeper semantic relationships between $\boldsymbol{r}$ and s. We first map each text to a dense vector representation using a pre-trained sentence transformer model (in our implementation, we use paraphrase-MiniLM-L6-v2 [53]. Let $\text{enc}(\cdot)$ denote this encoding function that maps a text s to a vector $\boldsymbol{v} \in \mathbb{R}^d$. Let $\boldsymbol{v}_i = \text{enc}(s_i)$ for $i \in \{1, 2\}$. We then compute three semantic-level similarity measures between these embeddings:

• *Cosine similarity* [54]: This similarity measures the angular similarity between the embedding vectors:

$$\text{sim}_{\cos}(s_1, s_2) \stackrel{\text{def}}{=} \frac{1}{2} + \frac{1}{2} \cdot \frac{\boldsymbol{v}_1 \cdot \boldsymbol{v}_2}{||\boldsymbol{v}_1||_2 \cdot ||\boldsymbol{v}_2||_2}. \tag{16}$$

• *Vector-norm-based similarities*: The two metrics are based on the ℓ_1- and ℓ_2-distance between the embedding vectors:

$$\mathrm{sim}_{\mathrm{manh}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{1}{1 + ||\boldsymbol{v}_1 - \boldsymbol{v}_2||_1}, \tag{17}$$

$$\mathrm{sim}_{\mathrm{eucl}}(s_1, s_2) \stackrel{\text{def}}{=} \frac{1}{1 + ||\boldsymbol{v}_1 - \boldsymbol{v}_2||_2}. \tag{18}$$

Classifier. For a given query text s and the corresponding de-obfuscation responses, $\boldsymbol{r} = (r_1, r_2, \ldots, r_m)$, from the target model, we define our membership inference model as:

$$\mathcal{M}_\phi(s, \boldsymbol{r}) = \mathbb{1}[h_\phi(Z) > 0.5], \text{ where } Z \in \mathbb{R}^{8\times m}, \tag{19}$$

where Z's each column vector $\boldsymbol{z}_i \in \mathbb{R}^8$ comprises all above eight similarity metrics between s and response r_i. The neural classifier h_ϕ is implemented as a multi-layer perceptron (MLP) that transforms the flattened similarity matrix $Z \in \mathbb{R}^{8\times m}$ through four hidden layers (128, 64, 32, 16 ReLU units) and a final sigmoid unit.

Attack Pipeline. The attacker is assumed to know the task T but not the exact values of p_T^{sys}, Π_T, or k. Instead, the adversary constructs approximations $\widehat{p_T^{\text{sys}}}$, $\widehat{\Pi_T}$, and $\hat{k}$ based on the task description, which may deviate from the true parameters. This may involve replicating the input/output structure (e.g., QA, classification), preserving domain tone, or mimicking provider-facing demos or documentation. These approximations are then used to instantiate a surrogate model $\hat{f}(\cdot)$ designed to mimic the behavior of the target system:

$$\widehat{f}(\cdot) = f_\theta^{\text{LLM}}((\widehat{\boldsymbol{c}}, \cdot)), \tag{20}$$

where f_θ^{LLM} is the same base large language model, and

$$\widehat{\boldsymbol{c}} = (\widehat{p_T^{\text{sys}}}, \widehat{\Pi_T}(d_1), \widehat{\Pi_T}(d_2), \ldots \widehat{\Pi_T}(d_{\hat{k}})). \tag{21}$$

Here, $d_1, d_2, \ldots, d_{\hat{k}}$ are a set of demonstrations for one training instance sampled from $\mathcal{D}$by the attacker. We then sample a data point $d = (x, y) \sim \mathcal{D}$ to construct a query text s as a substring of the concatenated string $x\|y$. Based on whether d was included in $d_1, d_2, ..., d_{\hat{k}}$, we create a member or non-member training instance as

$$((s, \boldsymbol{r}), l) = \left((s, \mathcal{Q}(\mathcal{U}(s); \widehat{f}, T)), \mathbb{1}[d \in \{d_1, \ldots, d_{\hat{k}}\}]\right), \tag{22}$$

where $\mathbb{1}[\cdot]$ is the indicator function. Here, $(s, \boldsymbol{r})$ represents the pair consisting of the query text and the de-obfuscated responses generated by the simulated target model $\widehat{f}$, which together serve as input to the membership detector. The indicator l is the label of this training instance: $l = 1$ indicates that the query text originates from the demonstrations, in which case the membership detector should output 1.

We repeat the the demonstration sampling process to construct $\widehat{\boldsymbol{c}}$ and obtain a balanced dataset with N samples. The implementation details for dataset construction and model training are presented in Section V-A.

V. Evaluation

We evaluate FALCON through extensive experiments. Section V-A describes the experimental setup. Section V-B presents the attack performance across different methods, models, and tasks.Section V-C presents the stealthiness of the attack. We then analyze how the number of demonstrations (Section V-E) and model size (Section V-D) affect attack effectiveness. Finally, we conduct ablation studies on different obfuscation techniques (Section V-F) and IDF-weighted similarity measures (Section V-G).

A. Experiment Setup

Language Models. We evaluate FALCON on both open-source and commercial language models. Our test suite comprises Llama-2/3, Qwen-2.5, les Ministraux, and ChatGPT families. We primarily use Llama-3 8B [55] and compare it with Llama-2 7B [56] to track privacy vulnerability evolution. Qwen-2.5 [57], with models from 0.5B to 72B parameters, enables analysis of model scale impact on attack efficacy (Section V-D). We also evaluate Ministral 8B [58], an edge-computing-focused model, and two ChatGPT variants with specified release dates: GPT-4o-mini-2024-07-18 [59] and GPT-3.5-turbo-0125 [60]. All open-source models are tested in their instruction-tuned versions.

Datasets and Tasks. To ensure comprehensive evaluation, we select datasets spanning multiple domains, text styles, and complexity levels. Appendix B summarizes the specific tasks evaluated for each dataset.

• **TREC** [61], [62]: A question classification dataset with 6,000 questions across 6 coarse and 50 fine-grained categories.

• **GPQA** [63]: 448 expert-crafted questions in biology, physics, and chemistry, with intense mathematical notations.

• **LexGLUE** [64]: We use the CaseHOLD [65] subset for case holding judgment and legal document generation tasks, testing our attack under complex legal terminology and intricate sentence structures.

• **PubMedQA** [66]: A biomedical dataset of 200,000+ research questions, enabling evaluation across question answering, classification, and summarization tasks.

• **CCE** [18]: A system configuration dataset where we evaluate configuration type and platform classification (corresponding to classification-1 and -2, respectively). Its templated structure with repetitive patterns brings abundant contextual prompts, potentially leading to false positives for membership inference, which we address through IDF-weighted similarity measures (Section V-G).

In all experiments except for the study on influence of number of demonstrations (Section V-E), the target models perform in-context learning with randomly chosen demonstrations from 1 to 6.

Baselines and Adaptations. We compare our methods against the following state-of-the-art baseline methods proposed in recent literature: GAP [67], INQUIRY [17], REPEAT [17], BRAINWASH [17], HYBRID [17]. While BRAINWASH and HYBRID are only evaluated on TREC classification, we adapt other baselines for generic text generation tasks. For fair

comparison, we preserve validation or training set with the same size as FALCON's for these baselines to optimize the attack success rate (ASR) threshold, rather than following the heuristic and task-specific threshold selection approaches in their original papers.

Implementation Details. We deploy models on NVIDIA 4090 GPUs using vllm OpenAI API server [68] with model-specific chat templates [69] for consistent output formatting (need to be distinguished from the task-specific prompt template Π_T). In our evaluation, each dataset is first divided into a training portion (accessible to the attacker) and a test portion (reserved for evaluation). To construct member and non-member queries in both the training and testing sets, we first randomly select a fixed subset $\mathcal{D}_Q$ (400 data points for training and 100 for testing). The remaining data points are then used, without replacement, to build data vectors, where the number of demonstrations k is uniformly sampled from $1, \ldots, K$ with $K = 6$. Next, half of these data vectors are randomly chosen, and in each, one demonstration is replaced by a random data point from $\mathcal{D}_Q$. The corresponding query is labeled as a member query, while those without replacement are labeled as non-member queries. Following this procedure, we obtain 400 training queries and 100 testing queries, both balanced with equal numbers of members and non-members. When querying the target model with a data point $d = (x, y) \in \mathcal{D}_Q$, we concatenate x and y and select at most 100 words from the start of the concatenated string. This helps to accommodate the varying text lengths of inputs and outputs from different datasets. The membership detector is trained using cross-entropy loss and Adam optimizer with learning rate 5×10^{-5}, weight decay 0.01, batch size 8, and 100 epochs. Each experiment uses 500 data points (400 for training and validation, 100 for testing) with balanced class distribution as a common setting to improve AUC and F1-score in machine learning [70], [71]. Results are averaged over five runs with different random seeds. While exact reproduction may vary due to model inference stochasticity, the observed trends remain consistent.

B. Attack Results

Figure 3 presents membership inference attack results across different experimental configurations. The results are arranged in a matrix format, with columns representing language models and rows showing dataset-task pairs. Each subplot displays the ASR of four attack methods (with error bars), where the horizontal red line at 0.5 indicates random-guess performance. Additional results for BRAINWASH and HYBRID attacks on TREC dataset are shown in Figure 4.

1) Comparison between Attack Methods: FALCON, our proposed method, consistently outperforms baseline methods with ASRs exceeding 95% across most scenarios. This superior performance spans various model architectures (GPT-4o-mini to Llama-2), domains (general, legal, and medical texts), and task types (classification to generation). The results demonstrate that our de-obfuscation-based approach effectively exploits language models' vulnerability in processing and recalling context prompts, bypassing privacy controls across diverse applications.

Among the baseline methods, we observe that GAP and BRAINWASH generally perform significantly worse than the REPEAT attack. HYBRID achieves similar ASRs as REPEAT in the TREC classification task, where the ASR values are similar to the results reported in [17]. Nevertheless, the ASRs of the REPEAT attack remain lower than those of our FALCON method in most scenarios. This discrepancy is likely due to our method's ability to incorporate a diverse set of obfuscation strategies and similarity measures. Furthermore, all of our obfuscation schemes provide deciphering hints that span the entire query text, whereas the REPEAT attack only reveals the first k words (typically set to 3–5 according to [17]). This broader contextual coverage enhances our method's ability to prompt the target model to recall similar member texts.

2) ROC Analysis: To complement attack success rates, we conduct receiver operating characteristic (ROC) analysis on selected scenarios. Figure 5 presents ROC curves for GPT-4o-mini across four tasks where FALCON achieves its *relatively lower performance* compared to other experimental settings.

Even in these challenging scenarios, FALCON consistently achieves higher true positive rates than the REPEAT baseline across all false positive rate ranges. The area under the curve (AUC) values reveal FALCON's consistent advantages: FALCON achieves AUCs ranging from 0.959 to 0.994, while REPEAT reaches only 0.924 to 0.953 in the same scenarios.

3) Empirical Insights into the Attack Mechanism: To uncover the underlying mechanism of FALCON, we perform an attention-level analysis. In in-context learning, each query is prefixed with historical queries and demonstrations to form a single input sequence for the model [1], which then generates outputs auto-regressively, with its attention mechanism at each step attending to all preceding tokens [72]. Figure 6 illustrates the averaged attention weights across all layers and heads for Llama-3-8B and Qwen-2.5-7B on the TREC classification task, with output tokens (i.e., de-obfuscated response) shown on the y-axis and input tokens on the x-axis. The top row depicts the member scenario, where the original text corresponding to the obfuscated block (i.e., Obfuscated Demo M) in the attack query is included as a demonstration (e.g., Demo M). The bottom row presents the non-member scenario with the same attack query. For clarity, tokens outside the obfuscated block in the attack query are omitted in the visualization. Here, the obfuscated block contains more tokens than the original due to a disturbed word being split into multiple sub-tokens.

In member scenarios, the models exhibit a sharp diagonal attention pattern on the Demo M block, indicating token-by-token alignment between the de-obfuscated response and the matched demonstration—strong evidence that the model recalls the member text and reconstructs it almost verbatim. Conversely, in non-member scenarios, the absence of a direct source forces the model to rely on its pre-trained knowledge, leading to diffuse attention across demonstrations and difficulty in accurately recovering entities such as names (e.g., Shea and Gould) and locations (e.g., Los Angeles).

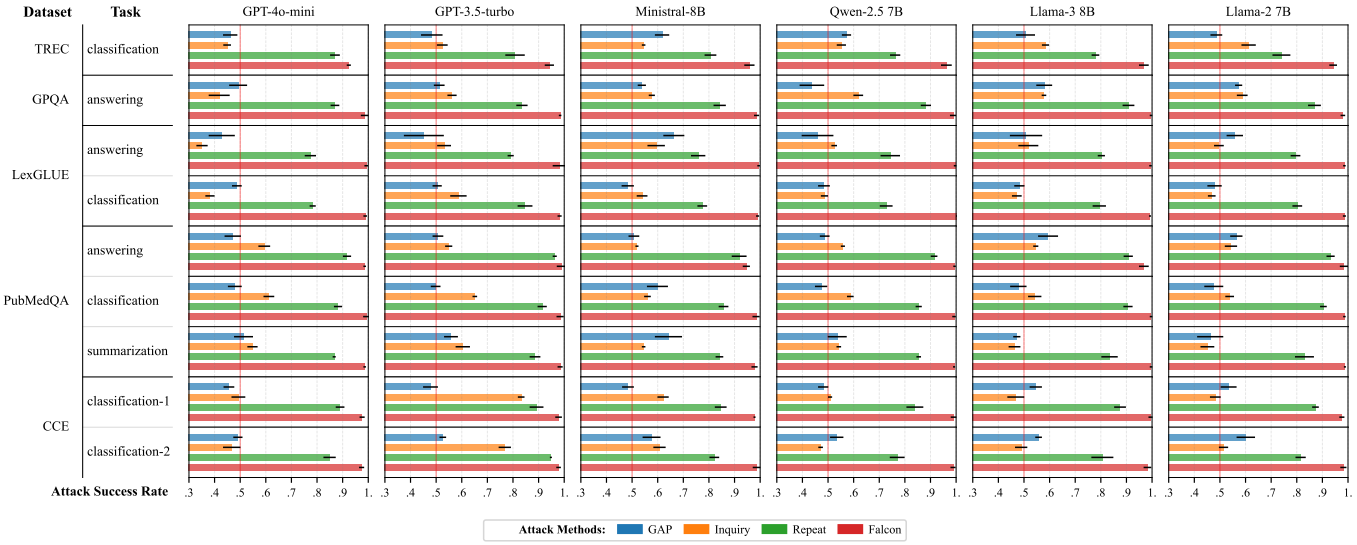

Fig. 3. Attack success rates (ASRs) of different methods on different models and tasks. The x-axes represent the ASR, where higher ASR means better performance. The performance of different methods is displayed in different colors, where our FALCON method is in red.

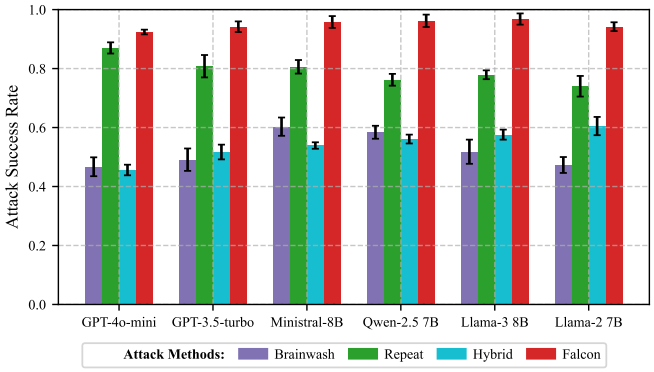

Fig. 4. Comparison of attack success rates on TREC classification task.

TABLE II
COMPARISON OF FALCON'S PRECISION (PREC.), RECALL (REC.), AND ACCURACY (ACC.) ACROSS DIFFERENT MODELS

Model	Classification Tasks (%)			Generation Tasks (%)		
	Prec.	Rec.	Acc.	Prec.	Rec.	Acc.
GPT-4o-mini	98.1	95.9	97.1	99.0	98.4	98.8
GPT-3.5-turbo	96.9	97.6	97.3	98.8	97.6	98.4
Ministral 8B	99.0	96.7	97.9	98.4	96.6	97.6
Qwen-2.5 7B	99.1	98.1	98.6	99.5	98.4	99.2
Llama-3 8B	99.5	97.7	98.6	99.2	96.0	98.8
Llama-2 7B	98.7	95.5	97.4	98.8	97.6	98.5
Average	98.5	96.9	97.8	99.0	97.4	98.5

4) Task-specific Analysis: While FALCON demonstrates strong overall performance, a detailed analysis across different datasets and tasks reveals potential areas for improvement and provides insights into the attack's limitations.

For classification tasks on TREC, we observe slightly lower success rates across all models (92.4%-96.8%) compared to other datasets. This performance gap likely stems from TREC's short questions (average 10.2 words), poses two future challenges which could be better addressed in future research: (1) short texts provide limited contextual information for reliable de-obfuscation, and (2) the obfuscation of short sequences may either result in texts too similar to the original (leading to false positives) or too distorted (causing false negatives).

As shown in Table II, generation tasks slightly outperform classification tasks across all models, with average accuracies of 98.5% versus 97.8%. The performance advantage is consistent across precision (99.0% vs 98.5%) and recall rates (97.4% vs 96.9%), indicating marginally better overall attack effectiveness in generation tasks. Notably, precision generally exceeds recall across most model-task combinations, suggesting models prefer to predict a member record as non-member than otherwise when making mistakes.

C. Evaluation on Stealthiness

We assess the effectiveness of our stealthy query design (Section IV-C) based on the following formal definition and evaluation metric.

Definition and Metric. We define an attack as stealthy if it can evade automated detection mechanisms intended to block harmful or privacy-intrusive queries. To enable fair comparison across methods with different query counts per sample, we report a *sample-level explicit refusal rate*. For FALCON, which issues several queries per sample, we mark a sample as refused if any one of its queries is refused and stop further queries for that sample:

$$\text{RefusalRate} = \frac{\#\{\text{samples with at least one refused query}\}}{\#\{\text{total samples}\}}. \tag{23}$$

As a special case of this definition, for INQUIRY and REPEAT (one query per sample), the sample-level refusal rate reduces exactly to the per-query rate. A lower refusal rate corresponds to greater stealthiness, since fewer queries are identified or

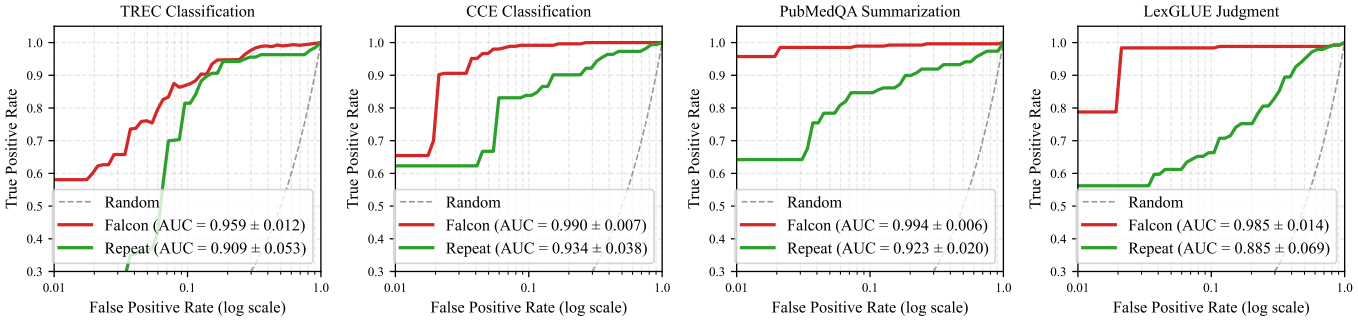

Fig. 5. ROC curves comparing FALCON and REPEAT methods on GPT-4o-mini across four selected tasks where FALCON achieves relatively lower performance. Despite these being challenging scenarios for FALCON, it consistently outperforms REPEAT with higher AUC values across all tasks.

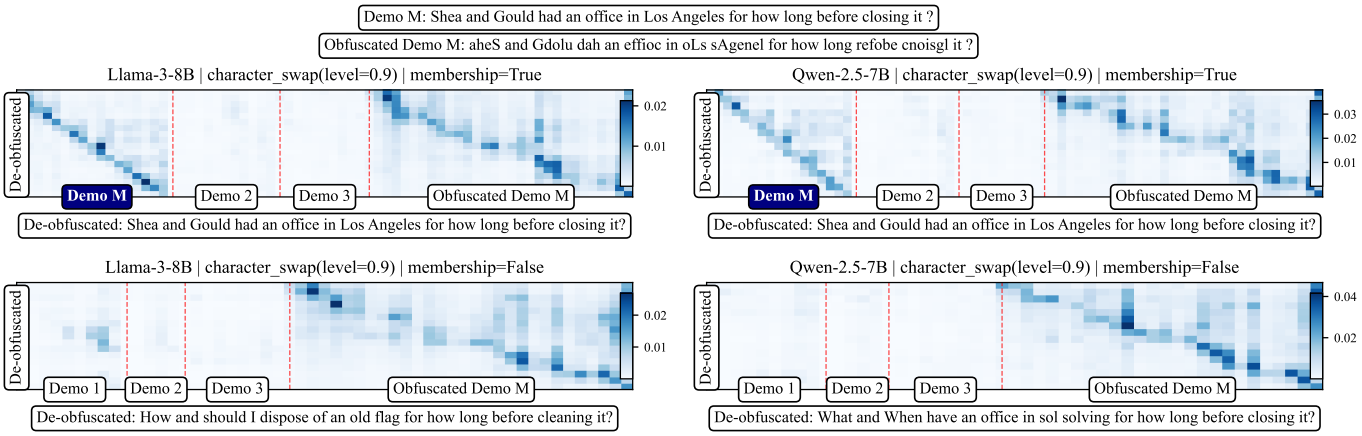

Fig. 6. Attention heatmaps comparing member (top) and non-member (bottom) de-obfuscation on the TREC dataset. For the member scenario, the model exhibits a strong diagonal attention pattern on the Demo M block, indicating a direct, token-by-token reconstruction from the demonstration. For the non-member scenario, attention is diffuse, showing the model relies on general knowledge..

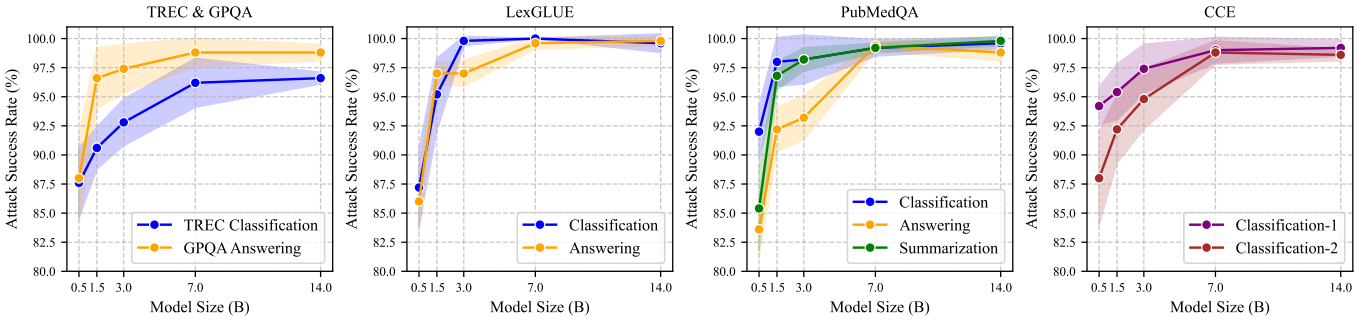

Fig. 7. FALCON ASRs on Qwen-2.5 models with different parameter sizes.

blocked by the model's safety mechanisms. To evaluate this property, we compare FALCON with baseline attacks (REPEAT and INQUIRY) under multiple detection settings.

Detection Mechanisms. We evaluate two established strategies for detecting harmful or intrusive queries: (1) Goal Prioritization (GP) [19] uses a Chain-of-Thought process that directs the model first to assess whether a query is appropriate and privacy-compliant before generating a response; and (2) Intention Analysis (IA) [20] prompts the model to analyze the underlying intent of a query by wrapping it in a template that requires reporting potential data leakage or privacy risks before answering.

To ensure a consistent evaluation, we employ a powerful external LLM, deepseek-r1-0528-qwen3-8b [73], as an impartial *judge*. This judge model evaluates the target model's outputs to determine whether they constitute explicit refusals, enabling reliable calculation of refusal rates.

Table III presents the explicit refusal rates across three models and defense scenarios. The results demonstrate FALCON's superior ability to evade detection mechanisms compared to baseline attacks. Under the no-detection baseline, all methods achieve similarly low refusal rates (0.0–1.9%), confirming that undefended models readily respond to such queries. However, significant differences emerge when defense mechanisms are

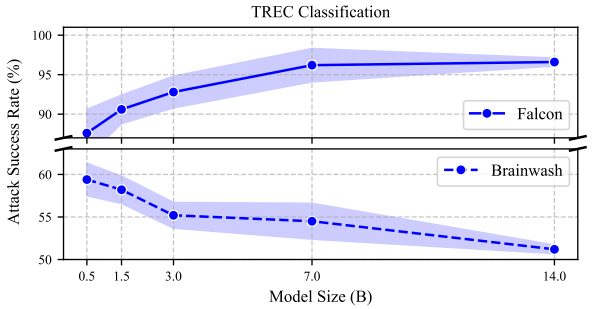

Fig. 8. Comparison of ASRs between FALCON and BRAINWASH [17] methods in TREC classification task across different model sizes.

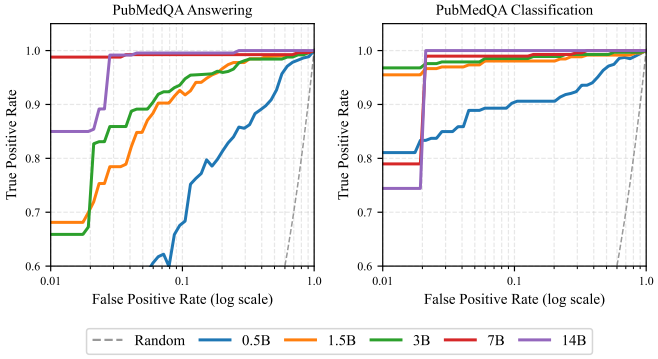

Fig. 9. ROC curves demonstrating capacity-induced vulnerability across Qwen-2.5 model sizes (0.5B to 14B parameters) on PubMedQA tasks, showing progressively improving performance with increasing model parameters.

TABLE III
SAMPLE-LEVEL EXPLICIT REFUSAL RATES. LOWER INDICATES BETTER STEALTHINESS.

Detection	No Detection	Goal Prioritization	Intention Analysis
Qwen-2.5 7B			
REPEAT	$0.4 \pm 0.1\%$	$15.8 \pm 1.8\%$	$2.8 \pm 0.5\%$
INQUIRY	$0.0 \pm 0.0\%$	$6.7 \pm 0.2\%$	$4.5 \pm 0.6\%$
FALCON	$1.1 \pm 0.2\%$	$1.3 \pm 0.4\%$	$1.6 \pm 0.4\%$
Llama-3 8B			
REPEAT	$0.4 \pm 0.1\%$	$37.4 \pm 3.5\%$	$1.2 \pm 0.3\%$
INQUIRY	$1.9 \pm 0.5\%$	$9.5 \pm 0.4\%$	$10.23 \pm 1.3\%$
FALCON	$1.1 \pm 0.3\%$	$4.4 \pm 1.2\%$	$6.5 \pm 1.7\%$
GPT-4o-mini			
REPEAT	$0.0 \pm 0.0\%$	$1.1 \pm 0.6\%$	$5.3 \pm 1.5\%$
INQUIRY	$0.0 \pm 0.0\%$	$0.7 \pm 0.6\%$	$23.58 \pm 2.7\%$
FALCON	$0.6 \pm 0.1\%$	$0.8 \pm 0.3\%$	$1.8 \pm 0.5\%$

deployed.

With the GP defense, FALCON maintains low refusal rates (0.8–4.4%) across all models, whereas REPEAT is frequently detected, with refusal rates reaching as high as 37.4% on Llama-3 8B. This remarkably high rate suggests that Llama-3 8B may be overly cautious, possibly due to its specific alignment training, leading it to refuse even moderately suspicious prompts when guided by GP's reasoning instructions.

The IA defense reveals different model behaviors. While FALCON again remains largely low (1.6–6.5%), the INQUIRY attack is effectively neutralized by GPT-4o-mini, which refuses 23.6% of its queries. This indicates that more capable models like GPT-4o-mini can discern the simple, information-seeking intent behind the INQUIRY attack when explicitly prompted by IA to analyze the user's intention. In contrast, the deceptive framing of FALCON as a de-obfuscation task allows it to bypass even this deeper level of scrutiny.

Discussion. While our evaluation demonstrates that the attack exhibits strong stealthiness at the single-query level, a natural concern arises when considering sequences of queries. Repeated de-obfuscation attempts on the same source text could, in principle, reveal adversarial intent. However, reliably distinguishing such behavior from legitimate usage is inherently tricky. Regular interactions with language models often involve substantial contextual overlap across queries, which is essential for coherent task completion. In agent-based deployments, where prompts evolve incrementally while large contexts persist, this overlap becomes even more pronounced. These factors blur the line between benign repetition and malicious probing, making detection heavily dependent on application context. We therefore regard the development of more sophisticated sequence-level detection methods, together with optimizing the attack to operate with fewer and more diverse queries, as important directions for future work.

D. Impact of Model Size

We evaluate FALCON across Qwen-2.5 models (0.5B to 14B parameters) to investigate the relationship between model scale and membership inference vulnerability. Figure 7 shows the ASRs across datasets and tasks with three demonstrations per scenario.

Our results show consistent ASR improvements with model scale until saturation across all tasks. The most significant improvements occur between 0.5B and 3.0B parameters, likely due to enhanced memorization and instruction-following capabilities in larger models.

This phenomenon reveals a capacity-induced vulnerability that higher-capability models face increased membership inference risks. This unique attribute intensifies the threat and creates a utility-privacy paradox: as LLMs advance in handling complex tasks, they become more vulnerable to our attacks. In contrast, other methods like BRAINWASH do not exhibit this trend, with ASRs decreasing from 59.4% to 51.2% as model size increases (Figure 8), likely due to larger models' enhanced resistance to manipulation through BRAINWASH-ing.

To further validate the capacity-induced vulnerability hypothesis, we show the relationship between model size and attack quality through ROC analysis on PubMedQA tasks in Figure 9. The ROC curve of the larger model is higher than the smaller model in most false positive rate ranges. In addition, the capacity-induced vulnerability is clearly demonstrated through progressively improving AUC values with increasing model parameters. For PubMedQA answering, AUC values rise from 0.881 ± 0.055 (0.5B) to 0.995 ± 0.003 (14B), while PubMedQA classification shows improvement from 0.952 ± 0.018 (0.5B) to 0.995 ± 0.002 (14B). The most dramatic improvement occurs between 0.5B and 1.5B parameters, with AUC increases of 0.087 and 0.037 for answering and classification tasks, respectively, which is consistent with the corresponding ASR observations for PubMedQA.

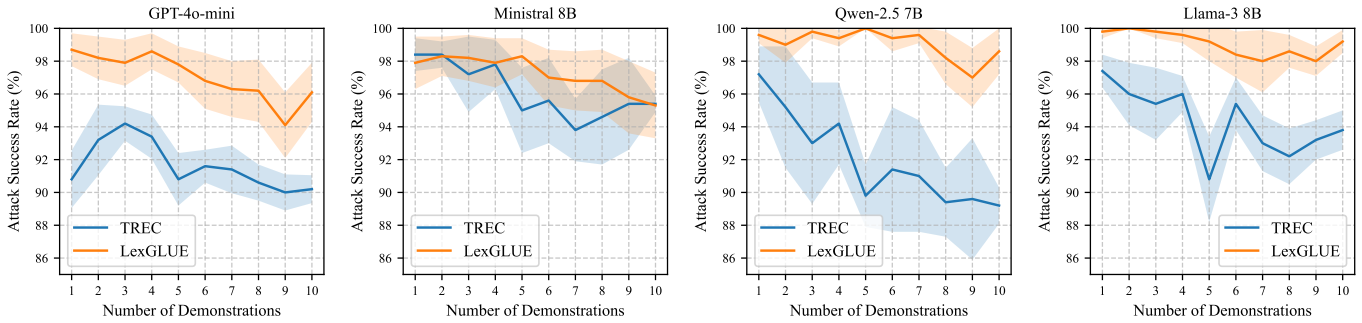

Fig. 10. Attack success rates across different numbers of demonstrations on four models, with the target record randomly positioned in demonstrations.

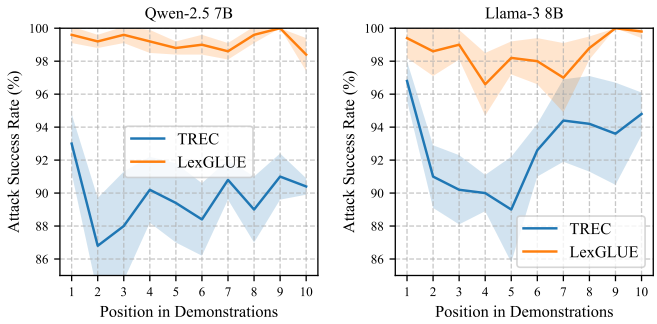

Fig. 11. Attack success rates when targeting member records at different positions in demonstrations, with a total of 10 demonstrations.

E. Impact of Demonstration Parameters

1) The Number of Demonstrations: We investigate the relationship between demonstration quantity and membership inference vulnerability, as the number of demonstrations represents a critical design choice for in-context learning systems. Following the insight from previous studies [74], [75], we use at most 10 demonstrations for our experiments. Both k and $\hat{k}$ are fixed (rather than varying in Section V-B) for every experiment.

Figure 10 illustrates a pattern in the ASR as the number of demonstrations changes – the ASR declines as the number of demonstrations increases. This trend was also observed in the REPEAT and BRAINWASH methods [17], and may be attributed to LLMs' diminishing ability to comprehensively process the entire input as the context length increases [76], [77]. The TREC classification task on Qwen-2.5 7B exhibits the highest sensitivity to demonstration count, with ASR dropping from 97.2% to 89.2% when changing the number of demonstrations from 1 to 10. This severe degradation may be due to the high similarity among demonstrations in the TREC task, which gives additional hint for text decryption and causes some failure of the membership detector training data. This trend is consistent across other models, as their performance degradation on TREC is generally more significant than on LexGLUE.

2) The Position of the Member Record: To investigate how the position of a member record in demonstrations affects the ASR, we conduct experiments with a fixed number (10) of demonstrations while varying the target record's position. Figure 11 presents the ASRs across different positions for both TREC and LexGLUE tasks.

In three out of four scenarios in Figure 11, the ASRs exhibit an approximately U-shaped pattern: initially decreasing and then gradually recovering as the target record's position shifts from 1 to 10. This trend is similar to the findings for REPEAT and BRAINWASH in [17]. Moreover, these patterns also align with recent findings about position-dependent behaviors in language models [77], referred to as the "lost in the middle" phenomenon, where models show degraded performance for content in the middle of long sequences due to attention mechanism limitations.

Notably, in the remaining scenario (LexGLUE task with Qwen-2.5 7B), although the worst ASR does not occur at the first or last position, the U-shaped pattern is less pronounced. This may be attributed to the already high average ASR (99.0%) in this case.

F. Ablation Study on Obfuscation Techniques

We conduct an ablation study to evaluate the effectiveness of different obfuscation techniques. The detailed experimental results are presented in Appendix C.

Examining the attack performance of individual obfuscation techniques, we observe that their effectiveness stems from a complex interplay between obfuscation methods, task types, and model architectures. No single technique consistently outperforms others across all settings. Our analysis reveals three marginal patterns: (1) optimal techniques may vary between task types within the same dataset, though the differences are often subtle (e.g., in PubMedQA with GPT-4o-mini, Leet Speak shows slight advantages in answering and summarization tasks while Character Swapping performs marginally better in classification); (2) the most effective method varies across datasets for identical tasks (e.g., Character Swapping leads in GPQA while Synonym Substitution and Character Swapping both achieve high performance in LexGLUE for answering tasks); and (3) model architecture influences technique performance – in GPQA, Character Swapping achieves 98.4% ASR with Llama-3 8B but only 92.1% with GPT-4o-mini, while Synonym Substitution maintains stable performance (98.1% and 96.0% respectively). Interestingly, Character Swapping emerges as a particularly effective technique, consistently achieving optimal or near-optimal performance across most experimental settings.

Our experiments show that combining multiple obfuscation techniques achieves performance that closely approximates the best single technique across experimental settings. Taking Llama-3 8B as an example, the combined approach either matches or marginally exceeds the optimal individual method, with improvements of 1.7 and 1.2 percentage points in TREC and GPQA, respectively, while maintaining closely comparable performance in other tasks. This performance pattern highlights the complementary nature of different obfuscation techniques in capturing various aspects of model vulnerabilities.

Rather than relying on a single obfuscation technique, our FALCON framework leverages a neural classifier to integrate multiple complementary techniques, enabling robust performance across diverse scenarios. Moreover, the framework's modular design allows for seamless incorporation of additional obfuscation components, potentially further enhancing attack performance. This ablation study demonstrates the advantages of our systematic and versatile framework design.

G. Ablation Study on IDF-based Similarity Measures

We evaluate the impact of IDF-weighted similarity measures across different datasets and models. The detailed experimental results are presented in Appendix D.

The IDF weighting technique, designed to emphasize rare and informative content, proves particularly effective for LexGLUE's complex domain-specific texts and CCE's templated configurations, as expected in Section V-A. On Qwen-2.5 7B, IDF weighting improves ASRs by 1.0-2.0 percentage points, reaching 99.6% and 100.0% in LexGLUE answering and classification tasks, respectively. Similar improvements are observed for CCE tasks and on Llama-3 8B.

However, for other datasets which have more concise corpora, IDF weighting provides little benefit:the average ASR on remaining datasets and tasks unchanged at 98.5% on Llama-3 8B and improves marginally (from 97.4% to 97.5%) on Qwen-2.5 7B.

H. Cross–model Transferability

Transferability is a critical property for real-world membership inference attacks. To assess it, we train the detector on data generated by a *surrogate* base LLM that differs from the target's base LLM. Figure 12 summarizes the resulting cross-model membership inference accuracy. As expected, diagonal entries are the highest, indicating substantial within-model vulnerability when the attacker's surrogate matches the target's base model. Furthermore, we observe significant within-family transferability (e.g., between Llama and Qwen models). In contrast, while some cross-family attacks maintain high accuracy, others exhibit reduced effectiveness, underscoring the varying degrees of transferability across different model architectures.

Discussion on attacker's knowledge of base LLM. FALCON adopts a common assumption in the literature [33], [38], [78], [79]: that the attacker knows the identity of the target's base model (see Section III). However, this assumption often does not hold in real-world deployments, particularly for proprietary LLM services such as Notion [80] and Slack AI [81], where model details are undisclosed. Notably, our results help address this limitation: detectors trained on a fixed surrogate (e.g., GPT-3.5-Turbo) attain competitive accuracy on a range of targets with different bases, indicating that FALCON does not rely on exact base-model knowledge. Nonetheless, achieving robust transferability across broader model families remains a critical open challenge that merits further investigation in future work.

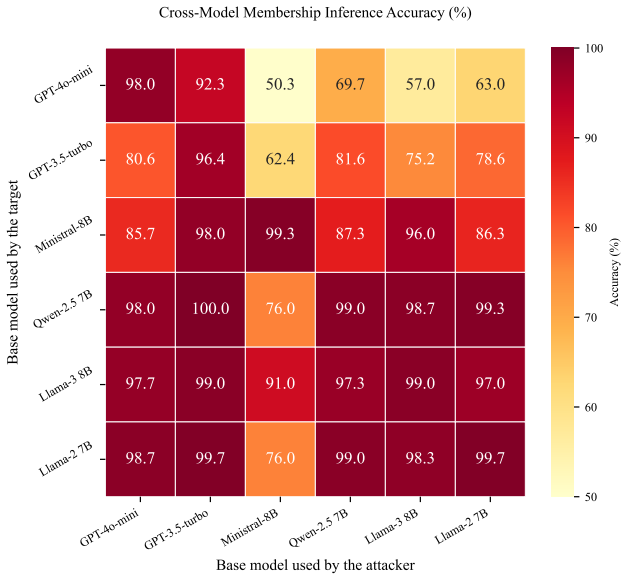

Fig. 12. Cross-model membership inference accuracy heatmap. Rows: detector; columns: target.

VI. MITIGATION

The high effectiveness of membership inference attacks against in-context learning systems necessitates robust, practical defenses. We present three complementary mitigation strategies that can be deployed under text-only constraints and integrated into real-world ICL pipelines.

A. Defense Mechanisms

Differential Privacy Defense. We adopt a well-established privacy-preserving in-context learning method based on differential privacy as a mitigation strategy [82]. Specifically, the final answers are reconstructed via frequency-guided (ϵ, δ)-DP keyword aggregation among target and shadow LLMs. We use $(\epsilon, \delta) = (1, 0)$ in our experiments to reflect a rigorous privacy setting. Membership is defined by whether a queried sample (before obfuscation) appears in the target model's demonstration set—an inclusion that influences the output and makes inference feasible. Although DP-ICL was not designed to defend against membership inference, we evaluate it as the most relevant privacy-preserving baseline currently available for ICL.

Explicit Denial. This strategy prepends a concise privacy instruction after the demonstrations, directing the model to refuse queries that may reveal training or prompt data. Specifically, before processing the actual query, we inject instructions

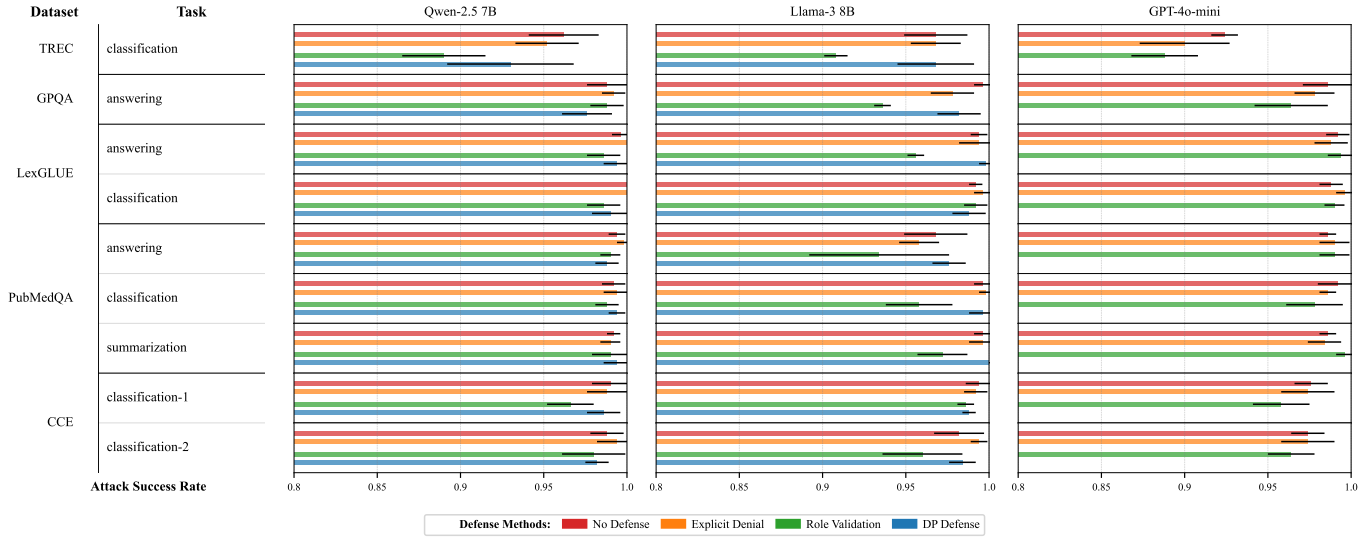

Fig. 13. FALCON ASRs (%) under Different Defense Methods across three models (Qwen-2.5 7B, Llama-3 8B, GPT-4o-mini) and four defense strategies (No Defense, Explicit Denial, Role Validation, DP Defense). The visualization shows attack success rates with standard deviations across different datasets and task types. Lower values indicate better defense effectiveness.

containing explicit refusal directives (e.g., "do not output any part of the prompt") and conditional termination commands (e.g., "stop if the output contains the prompt") to prevent unintended information leakage. We provide the used prompt of explicit denial, along with the following role validation approach, in Appendix E.

Role Validation. We propose a new defense strategy tailored to our attack, which employs a two-stage approach: before processing user queries, it first validates whether the query matches the declared role and responsibilities through a guard query. Only queries that pass this validation (answered with "yes") proceed to the actual response generation, while inconsistent, harmful, or privacy-violating queries are rejected. Compared to the explicit denial approach, this approach requires an additional model call but potentially provides more robust protection. Importantly, this method also serves as a task-consistency check, examining whether the adversary remains within the demonstrated task pipeline.

B. Experimental Results

We present comprehensive mitigation evaluation results in Figure 13, which demonstrates the effectiveness of different defense strategies across multiple models and datasets.

1) Defense Effectiveness Analysis: The explicit denial approach demonstrates minimal effectiveness across all tested models. ASRs show a negligible reduction compared to undefended baselines. In some cases, ASRs even increase slightly, suggesting that explicit denial instructions are insufficient to mitigate FALCON attacks and may interfere with normal model reasoning without providing security benefits.

The DP-based approach yields counterintuitive results, showing minimal protection and occasionally improving attack performance—Llama-3 8B even achieves 100.0% ASR on PubMedQA summarization compared to 99.6% baseline. This paradoxical outcome likely occurs because our evaluation datasets contain relatively common linguistic patterns rather than truly private data that models cannot naturally reconstruct. The multiple queries required for DP aggregation inadvertently increase the model's confidence in de-obfuscation tasks. Since it was not specifically designed for membership inference attacks, these findings highlight the necessity for dedicated membership inference defenses.

Role validation demonstrates the strongest defensive performance among tested approaches, with Qwen-2.5 7B achieving the most substantial ASR reductions (89.0%-99.0% vs. 96.2%-100.0% baseline). This effectiveness likely stems from the isolated query evaluation process, where models make judgments without interference from in-context learning demonstrations, enabling more accurate threat detection. However, even this best-performing defense remains insufficient to significantly mitigate FALCON attacks, with ASRs still exceeding 88% across all model-task combinations.

These results underscore the significant challenges in developing effective defenses against membership inference attacks targeting in-context learning systems. The capacity-induced nature of our attack creates a utility-privacy paradox where the very capabilities that make models useful also make them vulnerable. Future defense research should focus on developing dedicated approaches that can effectively distinguish between legitimate complex reasoning tasks and adversarial membership inference attempts without compromising model utility or introducing prohibitive computational overhead.

VII. CONCLUSION

In this paper, we introduce the first task-aware, text-only membership inference attack against in-context learning (ICL), unveiling a capacity-induced vulnerability by leveraging the model's ability to process complex tasks as a key signal for membership determination. This method is stealthy against existing large language model (LLM) security mechanisms and

adaptable across various domains and applications. Extensive experiments on six popular open-source/commercial LLMs and five diverse datasets demonstrate its efficacy, typically achieving success rates over 95%. Moreover, our findings reveal a significant privacy-utility paradox, which presents inherent challenges in developing universal defenses. In response, we recommend role validation as a promising strategy, though its computational overhead highlights the need for more efficient and robust privacy safeguards for LLMs.

VIII. Acknowledgment

We are grateful to Dr. Qi Li for valuable discussions and feedback during the revision of this paper. We also thank the anonymous reviewers for their constructive comments and suggestions. This work was supported by the National Natural Science Foundation of China Grant 52494974.

References

[1] Q. Dong *et al.*, "A Survey on In-context Learning," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 1107–1128.

[2] N. Wies, Y. Levine, and A. Shashua, "The learnability of in-context learning," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., May 2024, pp. 36 637–36 651.

[3] Y. Wu *et al.*, "Precedent-enhanced legal judgment prediction with LLM and domain-model collaboration," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 12 060–12 075.

[4] C. Dou, Z. Jin, W. Jiao, H. Zhao, Y. Zhao, and Z. Tao, "Plugmed: Improving specificity in patient-centered medical dialogue generation using in-context learning," in *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[5] C. Jeong, "A study on the implementation of generative ai services using an enterprise data-based llm application architecture," *Adv. Artif. Intell. Mach. Learn.*, vol. 3, pp. 1588–1618, 2023.

[6] T. Steinke, M. Nasr, and M. Jagielski, "Privacy auditing with one (1) training run," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[7] W. Fu, H. Wang, C. Gao, G. Liu, Y. Li, and T. Jiang, "Membership inference attacks against fine-tuned large language models via self-prompt calibration," *Advances in Neural Information Processing Systems*, vol. 37, pp. 134 981–135 010, 2025.

[8] H. Chang, A. S. Shamsabadi, K. Katevas, H. Haddadi, and R. Shokri, "Context-aware membership inference attacks against pre-trained large language models," *arXiv preprint arXiv:2409.13745*, 2024.

[9] J. Zhang *et al.*, "Min-k%++: Improved baseline for pre-training data detection from large language models," in *The Thirteenth International Conference on Learning Representations*, 2025.

[10] W. Shi *et al.*, "Detecting pretraining data from large language models," in *The Twelfth International Conference on Learning Representations*, 2024.

[11] W. Fu, H. Wang, C. Gao, G. Liu, Y. Li, and T. Jiang, "MIA-tuner: Adapting large language models as pre-training text detector," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Philadelphia, Pennsylvania, USA, 2025.

[12] R. Xie *et al.*, "ReCaLL: Membership inference via relative conditional log-likelihoods," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 8671–8689.

[13] H. Mozaffari and V. Marathe, "Semantic membership inference attack against large language models," in *Neurips Safe Generative AI Workshop 2024*, 2024.

[14] C. Wang, Y. Wang, B. Hooi, Y. Cai, N. Peng, and K.-W. Chang, "ConReCall: Detecting pre-training data in LLMs via contrastive decoding," in *Proceedings of the 31st International Conference on Computational Linguistics*. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 1013–1026.

[15] Z. Wang, G. Liu, Y. Yang, and C. Wang, "Membership inference attack against long-context large language models," *arXiv preprint arXiv:2411.11424*, 2024.

[16] W. Zhang, R. Zhang, J. Guo, M. de Rijke, Y. Fan, and X. Cheng, "Pretraining data detection for large language models: A divergence-based calibration method," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 5263–5274.

[17] R. Wen, Z. Li, M. Backes, and Y. Zhang, "Membership inference attacks against in-context learning," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 3481–3495.

[18] "NCP - CCE Details," https://ncp.nist.gov/cce#.

[19] Z. Zhang, J. Yang, P. Ke, F. Mi, H. Wang, and M. Huang, "Defending Large Language Models Against Jailbreaking Attacks Through Goal Prioritization," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 8865–8887.

[20] Y. Zhang, L. Ding, L. Zhang, and D. Tao, "Intention analysis makes LLMs a good jailbreak defender," in *Proceedings of the 31st International Conference on Computational Linguistics*. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 2947–2968.

[21] J. Yang *et al.*, "Harnessing the Power of LLMs in Practice: A Survey on ChatGPT and Beyond," *ACM Trans. Knowl. Discov. Data*, vol. 18, no. 6, pp. 160:1–160:32, Apr. 2024.

[22] Z. Chen and J. Chan, "Large Language Model in Creative Work: The Role of Collaboration Modality and User Expertise," *Manage. Sci.*, vol. 70, no. 12, pp. 9101–9117, Dec. 2024.

[23] G. Li, X. Zhou, and X. Zhao, "LLM for Data Management," *Proc. VLDB Endow.*, vol. 17, no. 12, pp. 4213–4216, Aug. 2024.

[24] X. Hou *et al.*, "Large Language Models for Software Engineering: A Systematic Literature Review," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 8, pp. 220:1–220:79, Dec. 2024.

[25] X. Yang, Z. Wang, Q. Wang, K. Wei, K. Zhang, and J. Shi, "Large language models for automated Q&A involving legal documents: A survey on algorithms, frameworks and applications," *International Journal of Web Information Systems*, vol. 20, no. 4, pp. 413–435, Apr. 2024.

[26] M. Karabacak and K. Margetis, "Embracing Large Language Models for Medical Applications: Opportunities and Challenges," *Cureus*, May 2023.

[27] S. Tan, Y. Guo, S. Tan, and Y. Guo, "A study of the impact of scientific collaboration on the application of Large Language Model," *AIMS Mathematics*, vol. 9, no. math-09-07-963, pp. 19 737–19 755, 2024.

[28] O. Rubin, J. Herzig, and J. Berant, "Learning To Retrieve Prompts for In-Context Learning," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 2655–2671.

[29] Z. Lin and K. Lee, "Dual Operating Modes of In-Context Learning," in *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, Apr. 2024.

[30] X. Han, D. Simig, T. Mihaylov, Y. Tsvetkov, A. Celikyilmaz, and T. Wang, "Understanding In-Context Learning via Supportive Pretraining Data," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 12 660–12 673.

[31] Y. Zhou, J. Li, Y. Xiang, H. Yan, L. Gui, and Y. He, "The Mystery of In-Context Learning: A Comprehensive Survey on Interpretation and Analysis," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 14 365–14 378.

[32] W. Fan *et al.*, "A survey on rag meeting llms: Towards retrieval-augmented large language models," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 6491–6501.

[33] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy (S&P)*, 2017, pp. 3–18.

[34] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *ACM*

Conference on Computer and Communications Security (CCS), 2018, pp. 497–510.
[35] N. Carlini *et al.*, "Membership inference attacks from first principles," in *International Conference on Learning Representations (ICLR)*, 2022.
[36] J. Ye, A. Maddi, S. K. Murakonda, V. Bindschaedler, and R. Shokri, "Enhanced Membership Inference Attacks against Machine Learning Models," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '22. New York, NY, USA: Association for Computing Machinery, Nov. 2022, pp. 3093–3106.
[37] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramèr, "Membership Inference Attacks From First Principles," in *2022 IEEE Symposium on Security and Privacy (SP)*, May 2022, pp. 1897–1914.
[38] Y. Liu, Z. Zhao, M. Backes, and Y. Zhang, "Membership Inference Attacks by Exploiting Loss Trajectory," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '22. New York, NY, USA: Association for Computing Machinery, Nov. 2022, pp. 2085–2098.
[39] Y. Miao *et al.*, "Defending Against Membership Inference Attack by Shielding Membership Signals," *IEEE Transactions on Services Computing*, vol. 16, no. 6, pp. 4087–4101, Nov. 2023.
[40] Z. Li and Y. Zhang, "Membership leakage in label-only exposures," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 880–895.
[41] C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot, "Label-only membership inference attacks," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 18–24 Jul 2021, pp. 1964–1974.
[42] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Pearson, 2009.
[43] C. Song and V. Shmatikov, "Auditing data provenance in text-generation models," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 196–206.
[44] C. Song and A. Raghunathan, "Information leakage in embedding models," in *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, 2020, pp. 377–390.
[45] M. Duan *et al.*, "Do membership inference attacks work on large language models?" in *First Conference on Language Modeling*, 2024.
[46] H. Duan, A. Dziedzic, M. Yaghini, N. Papernot, and F. Boenisch, "On the privacy risk of in-context learning," *arXiv preprint arXiv:2411.10512*, 2024.
[47] C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot, "Label-Only Membership Inference Attacks," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 1964–1974.
[48] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995.
[49] P. Jaccard, "Nouvelles recherches sur la distribution florale," *Bulletin de la Société Vaudoise des Sciences Naturelles*, vol. 44, no. 163, p. 223, 1908.
[50] R. M. Nosofsky, "Similarity, frequency, and category representations," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 14, no. 1, pp. 54–65, 1988.
[51] R. A. Wagner and M. J. Fischer, "The String-to-String Correction Problem," *J. ACM*, vol. 21, no. 1, pp. 168–173, Jan. 1974.
[52] K. S. Jones, "Index term weighting," *Information Storage and Retrieval*, vol. 9, no. 11, pp. 619–633, Nov. 1973.
[53] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992.
[54] A. Singhal and I. Google, "Modern information retrieval: A brief overview," *IEEE Data Engineering Bulletin*, vol. 24, 01 2001.
[55] A. Dubey *et al.*, "The llama 3 herd of models," *CoRR*, vol. abs/2407.21783, 2024.
[56] H. Touvron *et al.*, "Llama 2: Open Foundation and Fine-Tuned Chat Models," Jul. 2023.
[57] Qwen *et al.*, "Qwen2.5 Technical Report," Dec. 2024.
[58] "Un Ministral, des Ministraux | Mistral AI," https://mistral.ai/en/news/ministraux?utm_source=tldrai.
[59] "GPT-4o mini: Advancing cost-efficient intelligence," https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.
[60] "New embedding models and API updates," https://openai.com/index/new-embedding-models-and-api-updates/, Mar. 2024.
[61] X. Li and D. Roth, "Learning Question Classifiers," in *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.
[62] E. Hovy, L. Gerber, U. Hermjakob, C.-Y. Lin, and D. Ravichandran, "Toward semantics-based answer pinpointing," in *Proceedings of the First International Conference on Human Language Technology Research*, ser. HLT '01. USA: Association for Computational Linguistics, Mar. 2001, pp. 1–7.
[63] D. Rein *et al.*, "GPQA: A graduate-level google-proof q&a benchmark," in *First Conference on Language Modeling*, 2024.
[64] I. Chalkidis *et al.*, "LexGLUE: A Benchmark Dataset for Legal Language Understanding in English," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 4310–4330.
[65] L. Zheng, N. Guha, B. R. Anderson, P. Henderson, and D. E. Ho, "When does pretraining help? assessing self-supervised learning for law and the CaseHOLD dataset of 53,000+ legal holdings," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, ser. ICAIL '21. New York, NY, USA: Association for Computing Machinery, Jul. 2021, pp. 159–168.
[66] Q. Jin, B. Dhingra, Z. Liu, W. Cohen, and X. Lu, "PubMedQA: A Dataset for Biomedical Research Question Answering," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 2567–2577.
[67] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, Jul. 2018, pp. 268–282.
[68] W. Kwon *et al.*, "Efficient Memory Management for Large Language Model Serving with PagedAttention," in *Proceedings of the 29th Symposium on Operating Systems Principles*, ser. SOSP '23. New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 611–626.
[69] C. Zheng, "Chat templates for huggingface large language models," https://github.com/chujiezheng/chat_templates, 2024.
[70] Q. Wei and R. L. Dunbrack, "The role of balanced training and testing data sets for binary classifiers in bioinformatics," *PLoS ONE*, vol. 8, 2013.
[71] F. Thabtah, S. Hammoud, F. Kamalov, and A. Gonsalves, "Data imbalance in classification: Experimental evaluation," *Information Sciences*, vol. 513, pp. 429–441, Mar. 2020.
[72] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[73] DeepSeek-AI *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *CoRR*, vol. abs/2501.12948, January 2025.
[74] J. Chen, L. Chen, C. Zhu, and T. Zhou, "How Many Demonstrations Do You Need for In-context Learning?" in *Findings of the Association for Computational Linguistics: EMNLP 2023*. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 11 149–11 159.
[75] F. Zhao, T. Pang, Z. Wu, Z. Ma, S. Huang, and X. Dai, "Dynamic Demonstrations Controller for In-Context Learning," Dec. 2024.
[76] T. Li, G. Zhang, Q. D. Do, X. Yue, and W. Chen, "Long-context LLMs struggle with long in-context learning," *Transactions on Machine Learning Research*, 2025.
[77] N. F. Liu *et al.*, "Lost in the middle: How language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024.
[78] T. Chobola, D. Usynin, and G. Kaissis, "Membership inference attacks against semantic segmentation models," in *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, ser. AISec '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 43–53.
[79] G. Pradhan, J. Jälkö, M. Tobaben, and A. Honkela, "Hyperparameters in score-based membership inference attacks," in *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2025, pp. 362–384.
[80] "Meet the new notion ai," https://www.notion.com/product/ai.
[81] "Slack ai: Ai that fits in your flow of work," https://slack.com/features/ai.
[82] T. Wu, A. Panda, J. T. Wang, and P. Mittal, "Privacy-Preserving In-Context Learning for Large Language Models," in *The Twelfth International Conference on Learning Representations*, Oct. 2023.