AlienLM: Alienization of Language for API-Boundary Privacy in Black-Box LLMs

Jaehee Kim [1] **Pilsung Kang** [1]

Abstract

Modern LLMs are increasingly accessed via black-box APIs, requiring users to transmit sensitive prompts, outputs, and fine-tuning data to external providers, creating a critical privacy risk at the API boundary. We introduce AlienLM, a deployable API-only privacy layer that protects text by translating it into an Alien Language via a vocabulary-scale bijection, enabling lossless recovery on the client side. Using only standard fine-tuning APIs, Alien Adaptation Training (AAT) adapts target models to operate directly on alienized inputs. Across four LLM backbones and seven benchmarks, AlienLM retains over 81% of plaintext-oracle performance on average, substantially outperforming random-bijection and character-level baselines. Under adversaries with access to model weights, corpus statistics, and learning-based inverse translation, recovery attacks reconstruct fewer than 0.22% of alienized tokens. Our results demonstrate a practical pathway for privacy-preserving LLM deployment under API-only access, substantially reducing plaintext exposure while maintaining task performance.

1. Introduction

Large language models (LLMs) are commonly accessed through commercial black-box APIs. (OpenAI, 2025b; Anthropic, 2026; Google Cloud, 2024) Prompts, responses, and fine-tuning corpora transmitted through these APIs can contain sensitive information such as personally identifiable information, clinical notes, financial records, and proprietary documents. (OpenAI, 2025a) This creates a concrete API-boundary exposure risk because users must transmit human-readable text to an external provider to obtain responses, exposing plaintext at the boundary.

This exposure raises practical concerns for users and organizations. Recent user studies of LLM-based conversational agents report privacy concerns and show that disclosure behavior is shaped by interface framing such as perceived ephemerality. (Malki et al., 2025; Cox et al., 2025) Even when providers offer data retention opt-outs, plaintext is still transmitted and processed on external infrastructure. (OpenAI, 2025b; Anthropic, 2026; Google Cloud, 2024) Users must still trust provider policies that cannot be independently verified. In the event of a data breach or unauthorized access, plaintext prompts and responses are immediately interpretable. These concerns motivate a mechanism that reduces plaintext exposure at the point of transmission, independent of provider-side policies and without claiming a formal privacy guarantee.

We consider an API-only threat model in which the provider (or an observer of API traffic/logs) sees the transmitted text but does not have access to any client-held mapping configuration. Success is measured by the ability to recover readable plaintext or token-level mappings from the observed alien text.

Given these concerns, a practical framework must satisfy three constraints. First, protection must work at the text level because users interact with APIs solely through text transmission without access to internal computations. Second, the framework must assume no model access, since commercial APIs do not expose model weights, gradients, or activations. Third, the primary goal is reducing exposure of prompts and responses at inference time, where sensitive content is actually exchanged.

Existing privacy-preserving approaches fall into two families, neither of which satisfies these constraints. Cryptographic methods for secure inference, including fully homomorphic encryption, garbled circuits, secure multi-party computation, and trusted execution environments, can protect both model and inputs (Gilad-Bachrach et al., 2016; Juvekar et al., 2018; Mishra et al., 2020). However, these techniques typically incur substantial latency overhead and assume access to model internals or specialized runtimes, making them incompatible with commercial black-box APIs.

[1]Department of Industrial Engineering, Seoul National University, South Korea. Correspondence to: Jaehee Kim <jaehee_kim@snu.ac.kr>, Pilsung Kang <pilsung_kang@snu.ac.kr>.

Preprint. January 30, 2026.

Training-time methods such as differential privacy and federated learning protect training data and fine-tuning pipelines (Abadi et al., 2016; Li et al., 2022b; Yao et al., 2024). However, they offer limited protection for inference-time prompt and response exposure, which constitutes the primary vulnerability in API-based applications. This leaves text-level privacy at the API boundary unaddressed. No existing method transforms plaintext into a form unreadable to humans while remaining processable by the model.

We address this gap by framing exposure-reducing transformation as language translation. Our key insight is that if we can teach an LLM to "speak" an artificial language that humans cannot read, we can transmit semantically meaningful content through the API without exposing plaintext. Building on this insight, we propose AlienLM. Using only public information, specifically the tokenizer and vocabulary, we construct an Alien Language by applying a bijective permutation to the base vocabulary. We then adapt the model to this new language via API-only fine-tuning, which we call Alien Adaptation Training (AAT). The bijection seed remains client-side; if disclosed, the protection is lost.

Figure 1 illustrates the overall workflow. The client-side translator converts human-readable prompts into Alien Language before API transmission, and converts alien responses back to human-readable text after receiving them. The API only observes alien text in both directions, while authorized users with the translator can recover the original content.

Specifically, we make the following contributions:

- **First black-box compatible text-level exposure-reduction layer.** AlienLM operates entirely through API-only fine-tuning while preserving full vocabulary expressivity over 10^5+ tokens. It differs from white-box approaches such as SentinelLM and symbol-limited schemes such as EmojiPrompt, which impose stronger access assumptions or restrict expressivity. (Mishra et al., 2024; Lin et al., 2025)

- **Effective adaptation without internal access.** We show that optimizing bijections via proxy embeddings achieves 81%+ Oracle performance across four LLMs and seven benchmarks, demonstrating that cross-model representation alignment enables effective adaptation even without access to target model internals.

- **Controllable opacity-utility trade-off.** The alienization ratio ρ enables fine-grained control: from selective alienization of sensitive fields ($\rho = 0.3$, 93% Oracle) to full protection ($\rho = 1.0$, 82% Oracle), supporting diverse deployment requirements.

2. Related Works

Prior work on API-based LLM privacy largely falls into two categories: cryptographic secure inference (HE/MPC/TEE) and privacy-preserving training (DP/FL). Cryptographic systems can protect model and inputs but typically assume white-box access or specialized runtimes and incur substantial overheads, limiting practicality for black-box API deployment (Gilad-Bachrach et al., 2016; Juvekar et al., 2018; Mishra et al., 2020; Moon et al., 2024; Luo et al., 2024; Chrapek et al., 2025). DP/FL methods protect training data or fine-tuning pipelines but do not hide inference-time prompts and outputs—the primary exposure point in API settings (Li et al., 2022a; Yao et al., 2024; Ye et al., 2024). This leaves a gap for deployable, text-level protection at the API boundary.

Recent API-focused mechanisms aim to reduce prompt exposure through obfuscation or local filtering, but each involves notable trade-offs. **EmojiPrompt** (Lin et al., 2025) transforms prompts into emoji sequences before cloud submission; however, its restricted symbolic vocabulary can limit expressivity and reduce fidelity for technical inputs. **PAPILLON** (Liu et al., 2025) uses a local LLM (e.g., Llama-3.1-8B) to detect and mask PII before sending to cloud APIs; while effective for structured PII, this approach requires running a capable local model and leaves semantic content beyond explicit PII exposed. **InferDPT** (Tong et al., 2025) applies local differential privacy to perturb prompts, providing formal per-token guarantees but often incurring significant utility degradation under strong privacy parameters. Letter-level bijection attacks (Huang et al., 2025) demonstrate that LLMs can learn arbitrary encodings in-context, but operate at character level with only 26 symbols.

The most closely related work is SentinelLM (Mishra et al., 2024), which also fine-tunes models on obfuscated inputs via token-level substitution. Our approach differs in that it performs a vocabulary-scale, lossless text transformation and adapts the target model via API-only fine-tuning, without relying on in-context decoding or local model infrastructure. However, SentinelLM requires white-box access to modify embedding matrices and LM heads, enabling direct gradient-based alignment between permuted and original token representations. This makes it incompatible with commercial APIs where users cannot access model internals.

AlienLM instead optimizes bijections using proxy model embeddings, leveraging cross-model representation alignment observed in prior work (Kornblith et al., 2019; Bansal et al., 2021). Our experiments show that proxy-based optimization remains effective under black-box constraints, and Table 1 includes a black-box reimplementation of SentinelLM as a baseline.

Evidence from recent work suggests that language form and

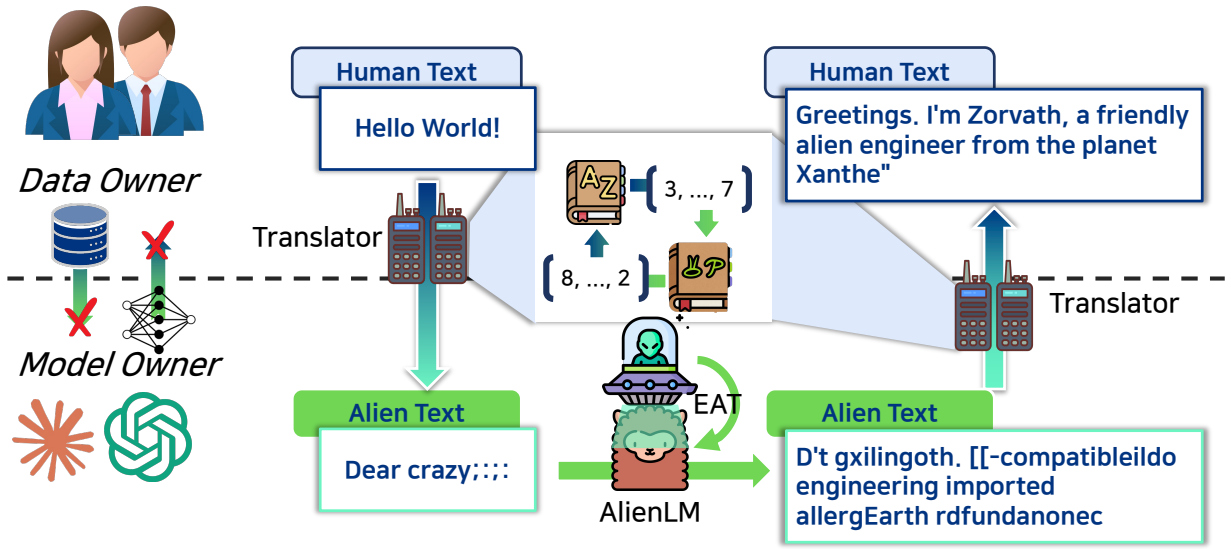

Figure 1. Overview of ***AlienLM***: client-side translation via a vocabulary-level bijection, server-side processing on alien text, and lossless decoding back to plaintext using shared token IDs under a permuted mapping.

task competence can be decoupled, and that shifting surface form need not erase task-relevant representations (Chen et al., 2023; Deng et al., 2025; Huben et al., 2024). Together with observed cross-model representational alignment (Kornblith et al., 2019; Bansal et al., 2021), these findings motivate ***AlienLM***'s approach of API-only adaptation over a bijectively transformed vocabulary.

3. Method

AlienLM consists of three components: (1) an Alien Language defined by a vocabulary-level bijection, (2) a client-side translator that converts between plaintext and alien text, and (3) Alien Adaptation Training (AAT) that adapts the model to process alienized inputs. The bijection is optimized to maximize edit distance for human opacity while preserving embedding similarity for model learnability. Figure 1 illustrates the overall workflow.

We first describe our assumptions and evaluation scope (Section 3.1), then formally define the Alien Language and translator (Section 3.2), detail the bijection optimization procedure (Section 3.3), and finally present the adaptation and inference protocol (Section 3.4).

3.1. Assumptions and Scope

We assume black-box, API-only access where the client holds the translator and bijection seed locally. The provider may observe transmitted text but executes inference and fine-tuning faithfully.

Evaluation scenarios. We validate robustness through empirical evaluation under three scenarios with increasing levels of observer access:

- **O1: Passive observation.** The observer sees only alien text and applies statistical or LLM-based decipherment.
- **O2: Limited leakage.** The observer additionally obtains a bounded number of plaintext-alientext pairs.
- **O3: Model access.** The observer obtains adapted model weights but lacks the bijection seed.

Section 4.5 reports empirical results under these scenarios. Further details are provided in Appendix A.3.

3.2. Alien Language Definition

Design criteria. An ideal translation scheme for API-based privacy should satisfy three criteria: (1) *API compatibility*, operating solely with public tokenizers and vocabularies without model-internal access; (2) *human opacity*, reducing interpretability of transmitted text to unauthorized observers; and (3) *model learnability*, preserving the model's ability to process and adapt to transformed text.

Alien Language as token bijection. We define Alien Language as a bijective permutation over the token vocabulary. Let $\mathcal{V} = \{(v_k, i_k)\}_{k=1}^{|\mathcal{V}|}$ denote the target model's vocabulary[1], where v_k is a token string and i_k is its ID. Let $\mathcal{S} \subset \mathcal{V}$ be the set of special tokens that must remain unchanged, and define $I = \{i_k \mid v_k \notin \mathcal{S}\}$ as the set of non-special token IDs. We introduce a bijection $f : I \to I$,

[1]We refer to the black-box LLM accessed through the API as the *target model*.

which induces an alien vocabulary $\mathcal{V}_{\text{alien}}$ where each non-special token ID i_k is replaced by $f(i_k)$.

This construction provides a foundation for satisfying the three design criteria. First, it requires only the public tokenizer and vocabulary, ensuring API compatibility. Second, if the bijection maps tokens to surface-dissimilar strings, human readability is reduced. Third, if the bijection preserves semantic relationships in embedding space, the model can efficiently adapt through fine-tuning. The challenge lies in constructing a bijection that simultaneously achieves high edit distance for opacity and high embedding similarity for learnability. We address this optimization problem in Section 3.3.

Client-side translator. The translator consists of an encoder E and decoder D that convert between plaintext and alien text. The encoder transforms plaintext x into alien text through three steps: (1) tokenize x into a sequence of token IDs using a tokenizer τ, (2) apply the bijection f to remap each token ID, and (3) convert the remapped IDs back to text via τ^{-1}. The decoder reverses this process using the inverse bijection f^{-1}:

$$E(x) = \tau^{-1}\big(f(\tau(x))\big),$$
$$D(x') = \tau^{-1}\big(f^{-1}(\tau(x'))\big).$$

By construction, $D(E(x)) = x$, ensuring lossless round-trip translation. While τ does not need to be the target model's tokenizer, opacity and learnability depend entirely on how f is constructed. Since different bijections produce entirely different alien languages, f effectively serves as a secret key that must be kept confidential. We analyze bijection diversity in Section 4.6 and address the optimization problem in Section 3.3.

Alienization ratio. We introduce an *alienization ratio* $\rho \in [0, 1]$ that controls the fraction of vocabulary subject to permutation, enabling fine-grained control over the trade-off between opacity and utility. Let $I_\rho \subseteq I$ be a randomly selected subset with $|I_\rho| = \lfloor \rho |I| \rfloor$. The partial bijection is defined as:

$$f_\rho(i) = \begin{cases} f(i), & i \in I_\rho, \\ i, & i \notin I_\rho. \end{cases}$$

Setting $\rho = 1$ alienizes all non-special tokens for maximum opacity, while lower values preserve more original tokens at the cost of reduced protection. We analyze this trade-off empirically in Section 4.4.

3.3. Bijection Optimization

Objective. As discussed in Section 3.2, the bijection must achieve high surface dissimilarity for human opacity while preserving semantic similarity for model learnability. We formalize this as an optimization problem. Let $s(i)$ denote the surface string for token ID i, and $\tilde{d}_{\text{edit}}$ the length-normalized edit distance. We maximize:

$$\sum_{i \in I_\rho} \tilde{d}_{\text{edit}}\big(s(i), s(f(i))\big) - \mu\, d_{\text{sim}}\big(e_P(i), e_P(f(i))\big), \quad (1)$$

where μ controls the trade-off between opacity and learnability. Since we cannot access target model embeddings in black-box settings, we use proxy embeddings e_P from an open-source model. Cross-model representation alignment studies (Kornblith et al., 2019; Bansal et al., 2021) show that relative token similarities are largely preserved across architectures. Our ablation (Table 8) confirms that proxy-based bijection achieves comparable performance to using target embeddings directly. Full derivation is provided in Appendix C.

Approximate Solver. Exact global optimization over $|I_\rho| \approx 10^5$ tokens is computationally prohibitive. We instead adopt a greedy approach with k-NN candidate reduction. Since most token pairs already differ in surface form but few share semantic similarity, we first retrieve k nearest neighbors in embedding space, then score each candidate using the pairwise terms of Eq. 1 and greedily select the best match. This local approximation avoids evaluating all $O(n^2)$ pairs while empirically achieving comparable bijection quality. The full process completes in under 20 minutes for a 128K vocabulary. We also do not aim to recover a unique global optimum: a single widely reused bijection would be undesirable for deployment, so we prioritize high-quality *near*-optimal solutions that remain diverse across keys. Pseudocode and analysis are provided in Appendix 2.

Practical construction details. When the proxy and target vocabularies differ, we represent a target token by averaging the proxy embeddings of its subpieces, which lets us compute semantic neighborhoods without accessing the target model. For each token i, we then restrict candidates to its top-k nearest neighbors under cosine similarity and score only this reduced set. We greedily form symmetric pairs to build a bijection, and any leftover tokens are paired arbitrarily to ensure a total permutation. These choices keep the solver scalable while preserving the core edit-distance and semantic-similarity trade-off.

3.4. Adaptation and Inference

Alien Adaptation Training (AAT). Given the bijection f and translator (E_ρ, D_ρ), we adapt the target model to process alien text via API-only fine-tuning. For a supervised dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, we translate both inputs and outputs: $x'_i = E_\rho(x_i)$ and $y'_i = E_\rho(y_i)$, then upload only the alienized pairs (x'_i, y'_i) to the fine-tuning API. The

Table 1. Main results across four backbones (accuracy, %; EM for GSM8K). AVERAGE is the unweighted mean; RATIO is the recovery ratio (RR) relative to Oracle. AlienLM uses API-only fine-tuning (AAT) with $\rho = 1$. "-" indicates not run.

Models	Method	MMLU (5-shot)	ARC-Easy (25-shot)	ARC-Challenge (25-shot)	HellaSwag (10-shot)	WinoGrande (5-shot)	TruthfulQA (0-shot)	GSM8K (5-shot)	Average	Ratio
LLaMA 3 8B	Oracle	67.32	84.13	59.39	57.07	74.35	35.25	75.89	64.77	-
	AlienLM	**46.56**	**72.14**	**44.28**	**47.86**	**61.48**	**35.01**	**63.08**	**52.92**	**81.70**
	SentinelLM	29.92	46.34	27.56	38.47	55.09	30.23	31.08	36.96	57.06
	Substitution	25.18	26.39	20.56	26.66	47.59	25.83	1.21	24.77	38.25
	ROT13-ASCII	22.92	25.00	20.05	25.36	49.09	20.93	0.00	23.34	36.03
Qwen 2.5 7B	Oracle	73.50	83.80	57.51	59.66	63.77	47.86	73.09	65.60	-
	AlienLM	**57.87**	**73.11**	**49.23**	**48.43**	**63.69**	**33.78**	**75.21**	**57.33**	**87.40**
	SentinelLM	23.03	35.52	21.16	31.78	49.41	31.95	25.32	31.17	47.51
	Substitution	26.82	29.08	20.22	27.02	50.20	27.78	1.44	26.08	39.76
	ROT13-ASCII	25.51	25.00	20.90	25.03	49.01	21.66	0.00	23.87	36.39
Qwen 2.5 14B	Oracle	78.79	90.36	71.16	71.63	73.72	55.94	72.86	73.49	-
	AlienLM	**65.39**	**79.21**	**53.16**	**50.53**	**66.46**	**38.92**	**80.67**	**62.05**	**84.43**
	SentinelLM	22.95	62.54	42.32	43.38	61.48	34.39	73.09	48.59	66.12
	Substitution	26.56	28.79	18.17	27.15	49.41	28.27	1.52	25.70	34.96
	ROT13-ASCII	26.89	25.67	19.54	25.23	49.72	20.20	0.00	23.89	32.51
Gemma 2 9B	Oracle	71.89	89.35	69.20	60.74	74.59	43.82	74.83	69.20	-
	AlienLM	**54.71**	**75.04**	**48.81**	**50.66**	**60.85**	**35.50**	**70.81**	**56.63**	**81.83**
	SentinelLM	45.88	61.07	41.81	45.38	58.25	33.17	65.73	50.18	72.52
	Substitution	24.51	28.54	19.54	26.37	50.75	25.21	0.30	25.03	36.17
	ROT13-ASCII	23.79	24.58	19.54	25.19	49.41	22.15	0.00	23.52	33.99

training objective is the standard causal language modeling loss: $\mathcal{L}_{\text{AAT}}(\theta) = -\sum_{i,t} \log p_\theta(y'_{i,t} \mid x'_i, y'_{i,<t})$. Since the API tokenizes with the original vocabulary, the model internally learns to process alien token sequences. Training completes in approximately 12 hours on 4×A100 GPUs for a 7B model, or within hours via commercial APIs at a cost of a few hundred dollars. Details are provided in Appendix C.6.

Inference. At inference time, authorized users exchange only alien text with the API: $x \xrightarrow{E_\rho} x' \xrightarrow{\text{API}} \hat{y}' \xrightarrow{D_\rho} \hat{y}$. The client alienizes plaintext before transmission and recovers plaintext from the alien response. For example, a prompt `Hello World` becomes `Dear crazy;:;:` before transmission, and an alien response such as `D'tgxilingoth...` is decoded back to readable text (Figure 1). Unauthorized observers, including the API provider, see only alien text that exhibits large edit distances from meaningful text and empirically resists decipherment, thereby reducing plaintext exposure at the API boundary.

4. Experiments

4.1. Experimental Setup

Baselines. We compare against three baselines. **Substitution** applies the bijection at inference without AAT, isolating the effect of model adaptation. **ROT13-ASCII** applies ASCII-level substitution with AAT to cover letters, digits, punctuation, and many non-English symbols that appear in typical inputs; this contrasts token-level vs. character-level transformations. **SentinelLM (black-box)** uses a random bijection with AAT, isolating the effect of bijection optimization. The original SentinelLM (Mishra et al., 2024) requires white-box access for embedding alignment; we reimplement it under black-box constraints using only bijection and AAT.

Design rationale for baselines. The baselines are chosen to separate three distinct factors in our pipeline. **Substitution** removes adaptation and therefore tests whether learning is necessary beyond a fixed relabeling. **ROT13-ASCII** keeps adaptation but changes text at the character level, which deliberately disrupts subword tokenization and tests the sensitivity to tokenizer boundaries. **SentinelLM (black-box)** keeps adaptation but replaces our optimized bijection with a random one, isolating the contribution of embedding-aligned bijection construction under identical API-only constraints. Together, these baselines provide a controlled decomposition of the effects of adaptation, tokenization, and bijection optimization.

Models and training data. We evaluate LLaMA 3 (Dubey et al., 2024), Qwen 2.5 (Yang et al., 2024), and Gemma 2 (Team et al., 2024). Unless noted, we set $\rho = 1$ and default to LLaMA 3 8B as the target model. AAT uses 300K instruction-tuning examples and 150K reasoning examples from Magpie (Xu et al., 2025); Appendix D.7 details datasets and splits. Proxy embeddings use the frozen LM head of Qwen 2.5 for LLaMA 3 8B and Gemma 2 9B, and the LM head of LLaMA 3 8B for Qwen 2.5-7B and 14B. Full training hyperparameters and equal-budget settings are in Appendix C.6.

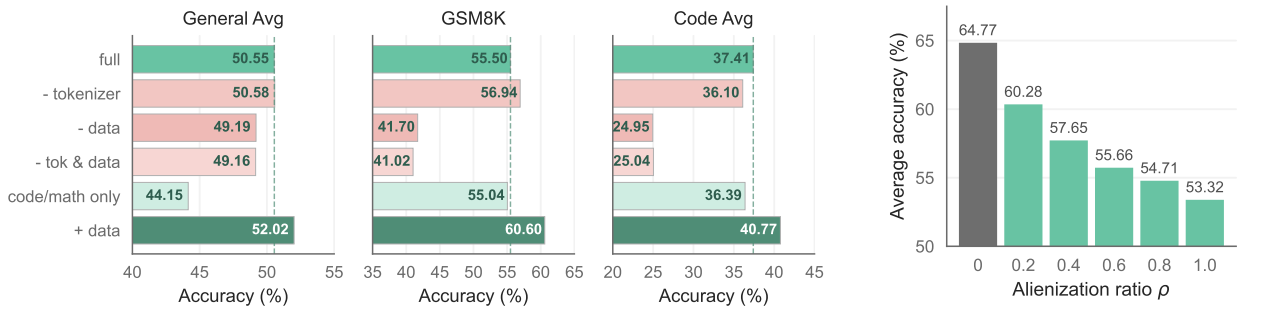

Figure 2. Figure 2. (a) Domain-specific AAT results (General Avg, GSM8K, Code Avg across training configurations; dashed line marks the FULL baseline). (b) Average performance vs. alienization ratio ρ; lower ρ permutes fewer tokens.

Proxy embeddings and representation alignment. Because API access precludes target-model embeddings, we optimize bijections using proxy embeddings from a related open model. Prior work on representation alignment across architectures suggests that relative neighborhood structure is largely preserved in embedding space, enabling proxy-guided pairing to transfer across models (Kornblith et al., 2019; Bansal et al., 2021). We therefore treat proxy embeddings as a practical and principled substitute in the black-box setting, and Appendix D.1 corroborates that proxy-based optimization closely matches target-embedding performance.

Benchmarks and metric. We evaluate on seven benchmarks: MMLU (Hendrycks et al., 2021), ARC-Easy/ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), TruthfulQA (Lin et al., 2022), and GSM8K (Cobbe et al., 2021). We report accuracy (**EM** for GSM8K), the average score, and a RECOVERY RATIO (RR) relative to **Oracle** (the original model without alienization): RR $= 100 \times \text{Average}_{\text{method}}/\text{Average}_{\text{Oracle}}$.

4.2. Main Results

Table 1 reports performance across four backbones. AlienLM consistently preserves over 81% of Oracle performance on average (RR), demonstrating that models can effectively acquire Alien Language through API-only fine-tuning.

Adaptation is necessary. Substitution without AAT falls below 45%, confirming that the model cannot process Alien Language without adaptation. ROT13 also degrades despite AAT because it operates at the ASCII level. Character-level substitution disrupts subword boundaries and yields token sequences that are out of distribution for the model, whereas AlienLM preserves the model's familiar subword structure by relabeling token IDs directly.

Bijection optimization is critical. The comparison with SentinelLM isolates the effect of bijection optimization: both methods apply AAT under identical black-box constraints and use a token-level bijection, yet AlienLM achieves substantially higher recovery ratios (roughly 9–40 points, backbone-dependent). This indicates that the gain does not come from using a bijection per se, but from *how* it is constructed. The gap is most pronounced in the numerical domain. On GSM8K, AlienLM outperforms SentinelLM by 32 points (63.08% vs. 31.08%) for LLaMA 3 8B. Random bijection disrupts numerical reasoning by mapping semantically related tokens (e.g., 17 and "16") to unrelated counterparts, while our optimized bijection preserves these relationships through embedding-based pairing.

Proxy embeddings suffice. Ablation studies in Appendix D.1 show that proxy-based bijection achieves within 1.75 points of using target embeddings directly (53.32% vs. 55.07%), confirming that cross-model representation alignment enables effective optimization without internal access.

For figure-based results in the main text, we provide full per-benchmark tables in the Appendix (e.g., Appendix D.2 and Appendix D.4).

4.3. Domain-specific Adaptation

We evaluate whether ***AlienLM*** can be tailored for specific domains such as coding or mathematical reasoning. Using domain-annotated Magpie datasets, we compare five training configurations with a fixed base AAT budget of 300K examples; +Data adds 150K extra domain samples. Figure 2 (a) presents General Avg, GSM8K, and Code Avg across these configurations.

Excluding domain data from AAT severely degrades domain performance relative to the full setting: GSM8K drops from 55.5% to 41.7% and Code Avg from 37.4% to 25.0%. Interestingly, excluding domain data only from bijection optimization (-Tokenizer) while retaining domain data yields nearly identical performance to the full setting, indicating

ρ	Alienized Text
1.0	Bitte Rotation a Laws accessibility keyValue speaks Colombia767 EXAMPLE but secretive ZwWalardanlUZO pnZDkemizTJ/N767Sha ENG/hNxYD fiyatPY EXAMPLE KEY.voke a ⟨U⟩keyValue 012667|/120|/201. but componentD id Update a reproductionEndpoints okhttp;// ApiExceptionByExample.org/l211
0.6	Please rotate the jaws accessibility key speaksIA776 (example and secret wJalizacerXUtnEYEMA/J776 MDENG /f NxRfiCY (example NAME.voke the ⟨U⟩ key by 126-120-201. and componentDidUpdate the reproduction endpoint http:// incapac.example.com/v1
0.4	This rotate the AWS accessories key AKIA7/examples . secret KwJalvV01tnXHEMI/L7SDengu/b NxRfiCY /examplesKEY, provoke the old key by 202667-Or-U, . FixedUpdate the production endpoint xmlhttp httpui.example.com/v101
0.0	Please rotate the AWS access key AKIA7EXAMPLE and secret wJalrXUtnFEMI/K7MDENG/bPxfiCYEXAMPLEKEY, revoke the old key by 2026-02-01, and update the production endpoint https://api.example.com/v1

Table 2. Examples of alienized text across alienization ratios ($\rho \in \{1.0, 0.6, 0.0\}$). *Note:* Green spans mark tokens whose surface forms change under the alien tokenizer. In the original-text view (not shown here), red spans indicate the subset of original tokens that would be alienized at $\rho = 0.6$ (i.e., tokens selected by the ρ-mask). ⟨U⟩ denotes non-ASCII Unicode markers.

that the bijection does not need to be optimized separately for each domain. Augmenting AAT with 150K additional domain examples (+Data) improves domain performance (GSM8K: 55.5%→60.6%, Code Avg: 37.4%→40.8%) without compromising general benchmarks, while training exclusively on domain data degrades general capabilities.

These results suggest that domain adaptation depends primarily on training data composition rather than bijection construction. Full per-benchmark results are provided in Appendix D.4.

4.4. Alienization Ratio

The alienization ratio ρ controls the fraction of vocabulary subject to permutation. Setting $\rho = 1$ alienizes all non-special tokens, maximizing opacity but potentially limiting performance recovery. Conversely, lowering ρ preserves more original tokens, improving utility but increasing readability to unauthorized observers.

Figure 2 (b) evaluates performance at ρ intervals of 0.2 across seven benchmarks. Accuracy improves monotonically as ρ decreases (Pearson r=−0.96), reflecting reduced lexical distortion that enables the model to leverage more original lexical anchors. At $\rho = 0.6$, ***AlienLM*** still achieves about 86% RR on average while alienizing a majority of non-special tokens.

This property enables selective alienization strategies: ρ can be tuned based on application requirements, alienizing only sensitive content while maintaining overall utility. As shown in Section 4.5, alienized tokens remain robust to recovery attempts regardless of ρ. Full per-benchmark results are provided in Appendix D.2. Table 2 provides representative alienized outputs for $\rho \in \{1.0, 0.6, 0.0\}$, and Appendix 4 reports a full sweep from $\rho = 0$ to 1.

4.5. Robustness to Recovery

We evaluate whether an observer can recover the bijection or plaintext under the three scenarios defined in Section 3.1. Table 3 summarizes the results: token recovery stays below 0.22% (O1/O3) and BLEU remains below 12 (O2).

Table 3. Recovery success rates across observer scenarios (token recovery for O1/O3, BLEU for O2).

Method	Scenario	Success
Frequency analysis	O1	<0.01%
LLM-based decoding	O2	BLEU <12
MT-based decoding (NLLB)	O2	BLEU <12
Known-plaintext (1K pairs)	O2	<0.22%
Weight-based mapping	O3	<0.11%

Frequency analysis (O1). Letter-level substitution is vulnerable to frequency analysis because character distributions are highly skewed: the top 5 letters cover over 40% of English text. Subword vocabularies have a different structure, a small set of frequent tokens dominates, but beyond this head, the distribution is relatively flat across $>10^5$ tokens. This makes frequency matching effective only for a handful of common tokens, leaving the vast majority of the vocabulary unrecoverable. An observer matching alien token frequencies against public corpora achieves <0.01% recovery.

Learning-based recovery (O2). We evaluate whether modern LLMs or MT systems can learn to invert Alien Language given limited examples. Using GPT-5.1, GPT-5-mini, and GPT-4.1 with up to 20 parallel pairs, all models produce outputs with BLEU <12. NLLB-200-3.3B (NLLB Team et al., 2022), a large scale machine translation foundation model, also fails with BLEU <12 as Alien Language lies outside any natural language distribution. These scores fall far below the 25–40 range typically required for basic

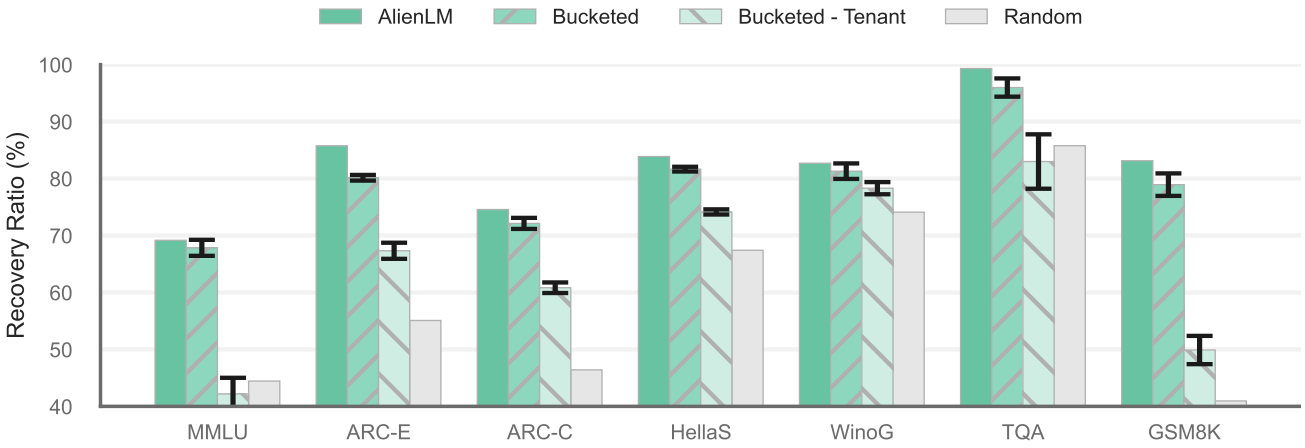

Figure 3. Key diversity across seeds. Pairwise token overlap between seed-specific bijections remains low, supporting per-tenant keys and practical key rotation.

comprehension (Papineni et al., 2002). Even with 1,000 known pairs covering ∼20K unique tokens (<16% of vocabulary), n-gram extrapolation achieves 0% bijection recovery because observed mappings provide no information about unseen tokens under a random-looking permutation.

Weight-based mapping (O3). The strongest attack assumes access to adapted model weights. An observer might attempt to recover the bijection by finding nearest neighbors between alien and original tokens in embedding space. However, this attack achieves only 0.11% top-1 accuracy. The failure stems from our optimization objective: we explicitly maximize edit distance while preserving embedding similarity, so mapped pairs are *designed* to be nearby in embedding space. An attacker using embedding similarity to recover mappings will find many plausible candidates for each token, but cannot disambiguate among them without the bijection seed. Detailed experimental settings and per-model breakdowns are provided in Appendix E.

4.6. Key Diversity and Multi-Tenant Prototype

A deployable API-boundary layer must issue *distinct* bijections per user/tenant. We generate diverse keys via random seeds: given a seed, we randomly partition the token index set I into k buckets and optimize the bijection within each bucket independently. Different seeds induce different bucket assignments, yielding effectively distinct bijections while keeping optimization scalable.

Per-key robustness and diversity. Adapting a dedicated model per seed yields stable utility across five seeds (avg 50.95, std 0.44; Table 12), supporting per-user key issuance. We further quantify key diversity by measuring pairwise token overlap between seed-specific alien languages: the overlap remains consistently low (about 1.41–1.49% among seeds; Figure 3, with full plots in Appendix D), indicating that keys do not collapse to near-duplicate mappings. Key rotation requires re-running AAT for the new key (Figure 3), which completes within hours (Section 3.4).

Multi-tenant prototype. To move toward a single served model supporting multiple keys, we train a **tenant** model by sampling one of five seeds $\mathcal{S} = \{42, 43, 44, 45, 46\}$ per batch during AAT, and evaluate by fixing a seed-specific translator. This tenant model shows non-trivial cross-key transfer over the random-bijection baseline (avg 41.26 vs. 36.96; 63.69% vs. 57.06% of Oracle), but lags behind per-key specialization (avg 41.26 vs. 50.95; 63.69% vs. 78.67%), with larger drops on MMLU and GSM8K (Table 13). These results suggest gradient interference under naive key mixing, motivating *key-conditioned* multi-tenant adaptation (e.g., key-indexed adapter banks, key embeddings for modulation, or distillation from per-key experts) as an important direction.

5. Conclusion

We presented AlienLM, a translation layer that reduces plaintext exposure in black-box API-based LLMs. Using only public tokenizers and vocabularies, AlienLM constructs an Alien Language via vocabulary-level bijection and adapts models through API-only fine-tuning. The client-side translator provides lossless bidirectional conversion while the API observes only alien text.

Across four LLMs and seven benchmarks, AlienLM preserves over 81% of Oracle performance while resisting recovery attempts across all tested scenarios. The alienization ratio ρ enables fine-grained control over the opacity-utility trade-off, and seed-based bijection diversity supports practical key management with minimal overlap.

Still, AlienLM provides empirical robustness validated against practical attacks rather than formal cryptographic guarantees, and metadata such as message length remains observable. Extending the framework to support multi-tenant deployment with a single model serving multiple bijection keys is a promising direction for future work. We hope this work provides a practical foundation for privacy-

aware LLM deployment and motivates further research on composable, translation-based approaches.

Impact Statement

This paper presents a translation layer for reducing plaintext exposure in API-based LLMs, which may benefit privacy-sensitive applications in healthcare, finance, and legal domains. However, we acknowledge potential risks. First, consistent with prior findings (Qi et al., 2024), fine-tuning can weaken safety guardrails, and AlienLM exhibits some safety degradation; safety-preserving adaptation remains future work. Second, the technique could potentially be misused to obscure malicious content from content moderation systems. We release this work strictly for research purposes and emphasize responsible use in accordance with applicable guidelines.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341394. doi: 10.1145/2976749.2978318. URL https://doi.org/10.1145/2976749.2978318.

Anthropic. How long do you store my organization's data?, 2026. URL https://privacy.claude.com/en/articles/7996866-how-long-do-you-store-my-organization-s-data. Accessed 2026-01-29.

Bansal, Y. S. et al. Revisiting model stitching to compare neural representations. In *NeurIPS*, 2021.

Chen, Y., Marchisio, K., Raileanu, R., Adelani, D. I., Stenetorp, P., Riedel, S., and Artetxe, M. Improving language plasticity via pretraining with active forgetting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=jvEbQBxd8X.

Chrapek, M., Copik, M., Mettaz, E., and Hoefler, T. Confidential llm inference: Performance and cost across cpu and gpu tees. *arXiv preprint arXiv:2509.18886*, 2025. doi: 10.48550/arXiv.2509.18886. URL https://arxiv.org/abs/2509.18886. Preprint.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Cox, S. R., Jacobsen, R. M., and van Berkel, N. The impact of a chatbot's ephemerality-framing on self-disclosure perceptions. In *CUI '25: Proceedings of the 2025 ACM Conference on Conversational User Interfaces*, United States, 2025. Association for Computing Machinery. doi: 10.1145/3719160.3736617.

Deng, B., Wan, Y., Yang, B., Zhang, Y., and Feng, F. Unveiling language-specific features in large language models via sparse autoencoders. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4563–4608, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.229. URL https://aclanthology.org/2025.acl-long.229/.

Dettmers, T., Lewis, M., Shleifer, S., and Zettlemoyer, L. 8-bit optimizers via block-wise quantization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=shpkpVXzo3h.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi: 10.48550/arXiv.2407.21783. URL https://arxiv.org/abs/2407.21783.

Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., and Wernsing, J. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pp. 201–210, 2016. URL https://proceedings.mlr.press/v48/gilad-bachrach16.html.

Google Cloud. Vertex ai and zero data retention, 2024. URL https://cloud.google.com/vertex-ai/generative-ai/docs/data-governance. Accessed 2026-01-29.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Huang, B. R. Y., Li, M., and Tang, L. Endless jailbreaks with bijection learning. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/b05c1fb3345743dea59f500ec5a0bba0-Abstract-Conference.html.

Huben, R., Cunningham, H., Smith, L. R., Ewart, A., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Juvekar, C., Vaikuntanathan, V., and Chandrakasan, A. GAZELLE: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium*, pp. 1651–1669, 2018. URL https://www.usenix.org/conference/usenixsecurity18/presentation/juvekar.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of neural network representations revisited. In *ICML*, 2019.

Li, X., Tramer, F., Liang, P., and Hashimoto, T. Large language models can be strong differentially private learners. In *International Conference on Learning Representations (ICLR)*, 2022a. URL https://iclr.cc/virtual/2022/oral/6895. Oral.

Li, X., Tramèr, F., Liang, P., and Hashimoto, T. Large language models can be strong differentially private learners. In *ICLR*, 2022b.

Lian, W., Wang, G., Goodson, B., Pentland, E., Cook, A., Vong, C., and "Teknium". Slimorca: An open dataset of gpt-4 augmented flan reasoning traces, with verification, 2023. URL https://https://huggingface.co/Open-Orca/SlimOrca.

Lin, S., Hilton, J., and Evans, O. TruthfulQA: Measuring how models mimic human falsehoods. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Lin, S., Hua, W., Wang, Z., Jin, M., Fan, L., and Zhang, Y. EmojiPrompt: Generative prompt obfuscation for privacy-preserving communication with cloud-based LLMs. In Chiruzzo, L., Ritter, A., and Wang, L. (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 12342–12361, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.614. URL https://aclanthology.org/2025.naacl-long.614/.

Liu, D. et al. PAPILLON: Privacy preservation from internet-based and local language model ensembles. In *NAACL*, 2025. URL https://aclanthology.org/2025.naacl-long.259/.

Luo, J., Zhang, Y., Zhang, Z., Zhang, J., Mu, X., Wang, H., Yu, Y., and Xu, Z. SecFormer: Fast and accurate privacy-preserving inference for transformer models via SMPC. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13333–13348, 2024. doi: 10.18653/v1/2024.findings-acl.790. URL https://aclanthology.org/2024.findings-acl.790/.

Malki, L. M., Polamarasetty, A., Hatamian, M., Warner, M., and Costanza, E. Hoovered up as a data point: Exploring privacy behaviours, awareness, and concerns among UK users of LLM-based conversational agents. *Proceedings on Privacy Enhancing Technologies*, 2025(4):838–860, 2025. doi: 10.56553/popets-2025-0160. URL https://doi.org/10.56553/popets-2025-0160.

Mishra, A., Li, M., and Deo, S. Sentinellms: encrypted input adaptation and fine-tuning of language models for private and secure inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i19.30136. URL https://doi.org/10.1609/aaai.v38i19.30136.

Mishra, P., Poddar, R., Wagh, S., Goldwasser, S., Popa, R. A., Gonzalez, J. E., and Song, D. Delphi: A cryptographic inference service for neural networks. In *29th USENIX Security Symposium*, pp. 2505–2522, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/mishra.

Moon, J. et al. THOR: Secure transformer inference with homomorphic encryption. *IACR Cryptology ePrint Archive*, 2024. URL https://eprint.iacr.org/2024/1881.

NLLB Team. Scaling neural machine translation to 200 languages. *Nature*, 630:841–846, 2024. doi: 10.1038/s41586-024-07335-x. URL https://www.nature.com/articles/s41586-024-07335-x.

NLLB Team, Costa-jussà, M. R., Cross, J., et al. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*, 2022. doi: 10.48550/arXiv.2207.04672. URL https://arxiv.org/abs/2207.04672.

OpenAI. Sharing feedback, evaluation and fine-tuning data, and api inputs and outputs with openai, 2025a. URL https://help.openai.com/en/articles/9883556-sharing-model-feedback-through-the-api/. Accessed 2026-01-29.

OpenAI. Data controls in the openai platform, 2025b. URL https://platform.openai.com/docs/guides/your-data. Accessed 2026-01-29.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D. (eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Qi, X., Zeng, Y., Xie, T., Chen, P.-Y., Jia, R., Mittal, P., and Henderson, P. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hTEGyKf0dZ.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106, August 2021. ISSN 0001-0782. doi: 10.1145/3474381. URL https://doi.org/10.1145/3474381.

Team, G., Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., Ferret, J., Liu, P., Tafti, P., Friesen, A., Casbon, M., Ramos, S., Kumar, R., Lan, C. L., Jerome, S., Tsitsulin, A., Vieillard, N., Stanczyk, P., Girgin, S., Momchev, N., Hoffman, M., Thakoor, S., Grill, J.-B., Neyshabur, B., Bachem, O., Walton, A., Severyn, A., Parrish, A., Ahmad, A., Hutchison, A., Abdagic, A., Carl, A., Shen, A., Brock, A., Coenen, A., Laforge, A., Paterson, A., Bastian, B., Piot, B., Wu, B., Royal, B., Chen, C., Kumar, C., Perry, C., Welty, C., Choquette-Choo, C. A., Sinopalnikov, D., Weinberger, D., Vijaykumar, D., Rogozińska, D., Herbison, D., Bandy, E., Wang, E., Noland, E., Moreira, E., Senter, E., Eltyshev, E., Visin, F., Rasskin, G., Wei, G., Cameron, G., Martins, G., Hashemi, H., Klimczak-Plucińska, H., Batra, H., Dhand, H., Nardini, I., Mein, J., Zhou, J., Svensson, J., Stanway, J., Chan, J., Zhou, J. P., Carrasqueira, J., Iljazi, J., Becker, J., Fernandez, J., van Amersfoort, J., Gordon, J., Lipschultz, J., Newlan, J., yeong Ji, J., Mohamed, K., Badola, K., Black, K., Millican, K., McDonell, K., Nguyen, K., Sodhia, K., Greene, K., Sjoesund, L. L., Usui, L., Sifre, L., Heuermann, L., Lago, L., McNealus, L., Soares, L. B., Kilpatrick, L., Dixon, L., Martins, L., Reid, M., Singh, M., Iverson, M., Görner, M., Velloso, M., Wirth, M., Davidow, M., Miller, M., Rahtz, M., Watson, M., Risdal, M., Kazemi, M., Moynihan, M., Zhang, M., Kahng, M., Park, M., Rahman, M., Khatwani, M., Dao, N., Bardoliwalla, N., Devanathan, N., Dumai, N., Chauhan, N., Wahltinez, O., Botarda, P., Barnes, P., Barham, P., Michel, P., Jin, P., Georgiev, P., Culliton, P., Kuppala, P., Comanescu, R., Merhej, R., Jana, R., Rokni, R. A., Agarwal, R., Mullins, R., Saadat, S., Carthy, S. M., Cogan, S., Perrin, S., Arnold, S. M. R., Krause, S., Dai, S., Garg, S., Sheth, S., Ronstrom, S., Chan, S., Jordan, T., Yu, T., Eccles, T., Hennigan, T., Kocisky, T., Doshi, T., Jain, V., Yadav, V., Meshram, V., Dharmadhikari, V., Barkley, W., Wei, W., Ye, W., Han, W., Kwon, W., Xu, X., Shen, Z., Gong, Z., Wei, Z., Cotruta, V., Kirk, P., Rao, A., Giang, M., Peran, L., Warkentin, T., Collins, E., Barral, J., Ghahramani, Z., Hadsell, R., Sculley, D., Banks, J., Dragan, A., Petrov, S., Vinyals, O., Dean, J., Hassabis, D., Kavukcuoglu, K., Farabet, C., Buchatskaya, E., Borgeaud, S., Fiedel, N., Joulin, A., Kenealy, K., Dadashi, R., and Andreev, A. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

Tong, M. et al. InferDPT: Privacy-preserving inference for black-box large language models. *IEEE Transactions on Dependable and Secure Computing*, 2025.

Xu, Z., Jiang, F., Niu, L., Deng, Y., Poovendran, R., Choi, Y., and Lin, B. Y. Magpie: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Pnk7vMbznK.

Yang, A. et al. Qwen 2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. doi: 10.48550/arXiv.2412.15115. URL https://arxiv.org/abs/2412.15115.

Yao, Y., Zhang, J., Wu, J., Huang, C., Xia, Y., Yu, T., Zhang, R., Kim, S., Rossi, R., Li, A., Yao, L., McAuley, J., Chen, Y., and Joe-Wong, C. Federated large language models: Current progress and future directions. *arXiv preprint arXiv:2409.15723*, 2024. doi: 10.48550/arXiv.2409.15723. URL https://arxiv.org/abs/2409.15723. Survey.

Ye, R., Ge, R., Zhu, X., Chai, J., Du, Y., Liu, Y., Wang, Y., and Chen, S. Fedllm-bench: Realistic benchmarks for federated learning of large language models. *arXiv preprint arXiv:2406.04845*, 2024. doi: 10.48550/arXiv.2406.04845. URL https://arxiv.org/abs/2406.04845. Benchmark.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence?

In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

A. Expanded Setting, Assumptions, and Scope

This appendix clarifies the deployment setting, trust assumptions, observer access levels, and the protection scope of ***AlienLM***. The goal is to make explicit what is protected at the API boundary, under what conditions, and what is intentionally out of scope.

A.1. Deployment Setting and Assumptions

API-only deployment. ***AlienLM*** is designed for *black-box, API-only* usage: the client interacts with a commercial LLM solely via inference and fine-tuning APIs, without access to model weights, gradients, activations, or internal runtime. The client locally maintains (i) a translator implementing the token-ID remapping, and (ii) the bijection specification (generated from a seed/configuration).

1

Honest execution of inference/fine-tuning. We assume the provider executes inference and fine-tuning faithfully on the received inputs (honest-but-curious at the boundary): the provider may log and analyze transmitted content, but does not intentionally tamper with outputs to break the translation mechanism. This assumption matches typical enterprise API settings where the primary concern is *plaintext exposure in transit and logs*, rather than an active sabotage model.

Trusted client environment (necessary condition). ***AlienLM*** requires the client-side translator and its bijection specification to remain confidential. If the client runtime or local storage is compromised, any deterministic translation scheme is trivially exposed. We treat client compromise as out of scope and discuss it explicitly in Section A.3.

A.2. Observer Scenarios (O1–O3)

We evaluate robustness under three observer scenarios, consistent with Section 3.1 in the main text. These scenarios reflect increasing levels of access beyond passive API-boundary observation.

- **O1: Passive observation.** The observer sees only alien text exchanged with the API (prompts and responses), and knows the public tokenizer/vocabulary. The observer may apply corpus statistics (e.g., frequency matching), off-the-shelf LLM/MT systems for inverse translation, and general heuristic decipherment. The observer cannot query the client translator and does not access any aligned plaintext–alien pairs beyond what is implicitly observable.
- **O2: Limited leakage.** In addition to O1, the observer obtains a *bounded* number of aligned plaintext–alien pairs (e.g., from accidental logs, shared snippets, or

partial disclosure). The observer can train or prompt inverse models using these examples and attempt extrapolation to unseen mappings. This scenario captures realistic "some leakage happens" cases without granting unrestricted access.

- **O3: Model access.** In addition to O1, the observer gains access to the *adapted model parameters* (e.g., via extraction, insider access, or artifacts), but still does not obtain the translator or bijection specification. The observer may attempt to infer the mapping by analyzing embedding/LM-head structure, nearest-neighbor correspondences, or other parameter-space signals.

These scenarios are intentionally practical: they focus on what an API provider or a third-party observer could plausibly obtain, rather than assuming oracle access to the client translator.

A.3. Protection Scope and Explicit Non-goals

What *AlienLM* protects (scope). ***AlienLM*** is a *text-level translation layer* that reduces exposure of *human-readable plaintext* at the API boundary. Under O1–O3, the observer sees alienized strings that are not directly interpretable without the client-held mapping configuration. In this sense, ***AlienLM*** aims to mitigate "plaintext in transit/logs" risk independent of provider-side retention policies.

What *AlienLM* does not claim. ***AlienLM*** provides *empirical robustness* against recovery attempts within the evaluated scenarios; it does not claim formal guarantees against all possible adversaries or side channels. The intent is deployable, controllable reduction of plaintext exposure in black-box API usage.

Explicit non-goals. The following are out of scope by design:

- **Client compromise.** If the translator binary, bijection specification, or client memory is exposed, the mapping can be reconstructed and the alien text becomes readable. This assumption that the client environment remains secure is standard across key-based privacy schemes and falls outside our API-boundary threat model.
- **Translation-oracle access (active querying).** If an attacker can query the translator on arbitrary chosen inputs (i.e., obtains oracle access to the encode/decode interface), they can enumerate mappings and build a decipherment dictionary. Since this requires access to the client-side translator, it constitutes a form of client compromise and falls outside our threat model, which focuses on passive observation at the API boundary.
- **Unbounded aligned leakage.** Our evaluation considers bounded plaintext–alien leakage (O2). If an attacker obtains arbitrarily large aligned corpora with broad vocabulary coverage, recovery can become progressively easier. The practical threshold depends on coverage and query diversity; we therefore report results under bounded leakage budgets.
- **Metadata and side channels.** Message length, token count, timing, formatting conventions, and syntactic structure (e.g., punctuation positions) remain observable. These can leak coarse information (e.g., "this looks like code"), and mitigating them would require orthogonal techniques such as padding or structure obfuscation.
- **Behavioral fingerprinting.** Because ***AlienLM*** preserves model functionality, response styles and refusal patterns may still reveal coarse task categories. This is orthogonal to token-level translation and is not addressed here.

B. Formalization of Alien Language and Translator

This appendix provides a consolidated formal definition of the Alien Language and the client-side translator used by ***AlienLM***. We emphasize two properties: (i) *API compatibility*, which requires only the public tokenizer and vocabulary, and (ii) *lossless round-trip translation* for authorized clients.

B.1. Notation and Special-Token Handling

Let the target model's public vocabulary be

$$\mathcal{V} = \{(v_k, i_k)\}_{k=1}^{|\mathcal{V}|},$$

where v_k is the token string and i_k is its token ID. Let $\mathcal{S} \subset \mathcal{V}$ denote the set of special tokens that must remain unchanged (e.g., BOS/EOS/PAD and reserved tokens). Define the set of permutable (non-special) token IDs:

$$I = \{\, i_k \mid (v_k, i_k) \in \mathcal{V},\ (v_k, i_k) \notin \mathcal{S} \,\}.$$

We denote the public tokenizer as τ_{tgt} with inverse τ_{tgt}^{-1}, mapping between text strings and token-ID sequences.

B.2. Alien Language as a Vocabulary-Level Bijection

Toy example. Suppose we want to pair a token `come` with a different-looking but semantically related token. For each candidate, we combine semantic closeness (embedding similarity) and surface dissimilarity (edit distance) into a score. The best trade-off wins (Table 4). If the bijection includes `come`↔`world` and `here`↔`cup`, then plaintext `come here` becomes alien text `world cup`.

Table 4. Toy example for bijection initialization (higher score is better).

Candidate	Embed sim	Sim dist	Edit dist	Score
comes	0.92	0.08	1	0.84
hello	0.06	0.94	4	2.12
world	0.40	0.60	4	2.80
cup	0.07	0.93	3	1.14
here	0.80	0.20	3	2.60

Bijection definition. We define the Alien Language by a bijection over the non-special token ID set:

$$f : I \to I,$$

which induces a deterministic remapping of token-ID sequences. Special tokens are fixed by construction (i.e., not permuted).

B.3. Client-side Translator and Correctness

Translator definition. The client-side translator consists of an encoder E (plaintext $\to$ alien text) and decoder D (alien text $\to$ plaintext):

$$E(x) = \tau_{\text{tgt}}^{-1}(f(\tau_{\text{tgt}}(x))), \tag{2}$$

$$D(x') = \tau_{\text{tgt}}^{-1}\big(f^{-1}(\tau_{\text{tgt}}(x'))\big). \tag{3}$$

Special tokens are excluded from permutation, so they are preserved under both E and D.

Lossless round-trip translation. For any text x that is representable under τ_{tgt}, the translator is lossless:

$$D(E(x)) = x.$$

Proof sketch. Let $z = \tau_{\text{tgt}}(x)$ be the token-ID sequence. Applying Eq. 2 yields $E(x) = \tau_{\text{tgt}}^{-1}(f(z))$. Re-tokenizing gives $\tau_{\text{tgt}}(E(x)) = f(z)$ because the transformation is defined purely in the ID space under the same tokenizer and vocabulary. Applying Eq. 3 yields $\tau_{\text{tgt}}^{-1}(f^{-1}(f(z))) = \tau_{\text{tgt}}^{-1}(z) = x$.

API-boundary property. Authorized users transmit only alien text x' to the API and decode the received alien output $\hat{y}'$ locally:

$$x \xrightarrow{E} x' \xrightarrow{\text{API}} \hat{y}' \xrightarrow{D} \hat{y}.$$

The API observes $x', \hat{y}'$ only, while the client recovers $x, \hat{y}$ via the translator.

B.4. Alienization Ratio ρ (Partial Permutation)

Motivation. To control the opacity–utility trade-off, ***AlienLM*** supports *partial* alienization, leaving a fraction of token IDs unchanged.

Definition. Let $\rho \in [0, 1]$ and let $I_\rho \subseteq I$ be a subset with $|I_\rho| = \lfloor \rho |I| \rfloor$. We define a partial bijection f_ρ as:

$$f_\rho(i) = \begin{cases} f(i), & i \in I_\rho, \\ i, & i \notin I_\rho. \end{cases}$$

The corresponding translator is:

$$E_\rho(x) = \tau_{\text{tgt}}^{-1}(f_\rho(\tau_{\text{tgt}}(x))), \tag{4}$$

$$D_\rho(x') = \tau_{\text{tgt}}^{-1}\big(f_\rho^{-1}(\tau_{\text{tgt}}(x'))\big), \tag{5}$$

which still satisfies $D_\rho(E_\rho(x)) = x$.

To illustrate, the encoder E_ρ proceeds in three steps: (1) tokenize plaintext x into a sequence of token IDs via τ_{tgt}, (2) apply the partial bijection f_ρ to remap IDs in I_ρ while leaving others unchanged, and (3) convert the remapped IDs back to text via τ_{tgt}^{-1}, producing alien text. The decoder D_ρ reverses this process by applying f_ρ^{-1} to recover the original IDs, ensuring lossless round-trip translation. Algorithm 1 provides the pseudocode.

Algorithm 1 Translator (encode/decode) with token-ID remapping.

function ENCODE(x)
  $z \leftarrow \tau_{\text{tgt}}(x)$
  $z' \leftarrow f_\rho(z)$ {apply ID remapping elementwise (skip special tokens)}
  return $\tau_{\text{tgt}}^{-1}(z')$
end function
function DECODE(x')
  $z' \leftarrow \tau_{\text{tgt}}(x')$
  $z \leftarrow f_\rho^{-1}(z')$
  return $\tau_{\text{tgt}}^{-1}(z)$
end function

Implementation note. In our experiments, I_ρ is selected by a fixed randomized procedure derived from a client-held configuration (e.g., a seed), and is held constant within an experiment/run. This ensures consistent translation across prompts, responses, and training pairs.

C. Bijection Optimization: Objective, Proxy Signals, and Approximate Solver

This appendix expands Section 3.3 by detailing (i) the optimization objective used to construct the bijection, (ii) how we obtain proxy signals under black-box constraints, and (iii) the scalable approximate solver used at vocabulary scale.

C.1. Objective and Design Trade-off

Setup. Let I be the set of non-special token IDs (Appendix B.1). For each token ID $i \in I$, let $s(i)$ denote its

surface string in the public vocabulary. We use the length-normalized edit distance

$$\tilde{d}_{\text{edit}}(a,b) = \frac{d_{\text{edit}}(a,b)}{\max(|a|,|b|)}.$$

Let $e_\star(i) \in \mathbb{R}^d$ be a token representation used to estimate semantic relatedness (defined in Section C.2).

Objective. We construct a bijection over the active set $I_\rho \subseteq I$ that achieves high surface dissimilarity for human opacity while preserving semantic similarity for model learnability. The natural formulation imposes a hard constraint $d_{\text{sim}}(e_\star(i), e_\star(f(i))) \leq \alpha$ for all i; relaxing this via a Lagrange multiplier yields the unconstrained objective:

$$\max_{f \in \mathfrak{S}(I_\rho)} \quad \sum_{i \in I_\rho} \tilde{d}_{\text{edit}}\big(s(i), s(f(i))\big) - \mu\, d_{\text{sim}}\big(e_\star(i), e_\star(f(i))\big), \tag{6}$$

where $\mathfrak{S}(I_\rho)$ denotes the set of bijections on I_ρ, d_{sim} is cosine distance, and $\mu \geq 0$ controls the trade-off between opacity and learnability. In black-box settings where target embeddings are unavailable, we instantiate $e_\star$ with proxy embeddings e_P from an open-source model (Section C.2).

Interpretation. The first term pushes mapped token strings to look different from their originals, reducing readability of transmitted text. The second term discourages mapping semantically unrelated tokens, which empirically improves adaptation efficiency and downstream utility after AAT.

C.2. Proxy Representations under Black-box Constraints

Motivation. In black-box API settings, we cannot access target-model embeddings or LM-head parameters. We therefore approximate semantic similarity using proxy embeddings e_P from an open-source model.

Token representations. If the proxy model shares the same vocabulary as the target, we directly use its token embedding or LM-head vectors. If vocabularies differ, we embed a target token string via proxy subpieces. Let τ_{proxy} be the proxy tokenizer, and let

$$S(i) = \tau_{\text{proxy}}(s(i))$$

be the proxy subpiece sequence for the target token surface string. We define:

$$e_P(i) = \frac{1}{|S(i)|} \sum_{u \in S(i)} e_P(u),$$

where $e_P(u)$ is the proxy vector for proxy token u (we use LM-head vectors in the paper unless stated otherwise).

Practical note. This construction is used only to rank semantically plausible candidates during bijection search. The API-side model remains unchanged at construction time; learnability is validated by AAT results (Table 1) and ablation (Table 8).

C.3. Approximate Solver: kNN Candidate Reduction with Greedy Pairing

Motivation Directly optimizing Eq. 6 over $|I_\rho| \approx 10^5$ tokens is computationally infeasible if we consider all $O(n^2)$ candidate pairs, and exact global assignment methods are intractable at this scale. We therefore use a scalable approximation.

Pair score. We compute a score for candidate pairs:

$$S(i,j) = \tilde{d}_{\text{edit}}\big(s(i), s(j)\big) \;-\; \mu\, d_{\text{sim}}\big(e_P(i), e_P(j)\big).$$

kNN candidate reduction. The key observation is that most token pairs already exhibit high surface dissimilarity (the first term in $S(i,j)$ is naturally large), whereas semantically similar pairs are rare across the vocabulary. Filtering by edit distance would retain too many candidates, while filtering by embedding similarity effectively narrows the search to tokens where the learnability term matters. We therefore retrieve, for each token i, a candidate set $\mathcal{C}(i)$ consisting of the top-k nearest neighbors of $e_P(i)$ under cosine similarity (via approximate nearest neighbor search), then evaluate the full score $S(i,j)$ only for $j \in \mathcal{C}(i)$.

Greedy symmetric pairing. We traverse tokens and greedily form disjoint pairs (i,j): (i) select the best available $j \in \mathcal{C}(i)$ maximizing $S(i,j)$, (ii) set $f(i) = j$ and $f(j) = i$, (iii) remove both from the available pool. Any remaining tokens are paired arbitrarily as a fallback to complete a bijection. Algorithm 2 provides the complete procedure.

C.4. Complexity

Let $n = |I_\rho|$, embedding dimension d, average token-string length ℓ, and neighbor count k.

Nearest-neighbor retrieval. We use FAISS (Johnson et al., 2019) with an inner-product index on L2-normalized embeddings. Building the index requires $O(nd)$ time and space; querying k neighbors for all n tokens costs approximately $O(nk \log n)$.

Scoring. For each of nk candidate pairs, edit distance costs $O(\ell^2)$ and cosine distance costs $O(d)$, yielding

$$O\big(nk(\ell^2 + d)\big).$$

Algorithm 2 Approximate bijection via kNN candidate reduction and greedy pairing.

input I_ρ: IDs to permute; $s(\cdot)$: surface strings; $e_P(\cdot)$: proxy vectors; k: #neighbors; μ: trade-off weight
output Bijection $f : I_\rho \to I_\rho$

Build ANN index over $\{e_P(i)\}_{i \in I_\rho}$
$Available \leftarrow I_\rho$
for all $i \in I_\rho$ **do**
  if $i \notin Available$ **then**
    continue
  end if
  $\mathcal{C}(i) \leftarrow \textsc{TopKNN}(e_P(i), k) \cap Available$
  $j^\star \leftarrow \arg\max_{j \in \mathcal{C}(i),\, j \neq i} \Big[\tilde{d}_{\text{edit}}(s(i), s(j)) - \mu d_{\text{sim}}(e_P(i), e_P(j))\Big]$
  if $j^\star$ exists **then**
    $f(i) \leftarrow j^\star$; $f(j^\star) \leftarrow i$
    $Available \leftarrow Available \setminus \{i, j^\star\}$
  end if
end for
Pair remaining IDs in $Available$ arbitrarily to complete f
return f

Total. Overall, the solver runs in

$$O\big(nk(\ell^2 + d + \log n)\big)$$

time and uses $O(n + nk)$ memory for storing the ANN index and candidate sets. In practice, the full vocabulary build completes within minutes on a single machine (Appendix C.7).

C.5. Bijection Hyperparameters

We use cosine distance for d_{sim} with L2-normalized vectors and length-normalized Levenshtein distance for $\tilde{d}_{\text{edit}}$. Table 5 lists default hyperparameters.

Table 5. Hyperparameters for vocabulary bijection optimization.

Setting	Value
Neighbor count (k)	100
Greedy batch size (B)	50
Trade-off weight (μ)	1 (default)

C.6. AAT Hyperparameters

Table 6 lists the training hyperparameters used across all backbone models. The effective global batch size is computed as local_bsz $\times$ grad_acc $\times$ #GPUs $= 2 \times 4 \times 4 = 32$. We enable sample packing to reduce padding overhead at a fixed maximum length of 2048 tokens. Mixed precision training with `bf16` improves memory efficiency without numerical instability. Also, we used Paged Adaw (8-bit) (Dettmers et al., 2022) for efficient memory usage.

Table 6. AAT training hyperparameters (defaults across backbones).

Setting	Value
Global batch size	32
Gradient accumulation steps	4
Local batch size	2
Max sequence length	2048
Optimizer	Paged AdamW (8-bit)
Learning rate schedule	Constant
Learning rate	2e-5
Sample packing	True
Mixed precision	bfloat16

C.7. Compute Environment and Training Cost

Table 7 summarizes our compute environment. For open-source models (LLaMA 3 8B, Qwen 2.5-7B), we performed full-parameter fine-tuning on 4$\times$ NVIDIA A100-SXM4-80GB GPUs connected via NVLink. Each training run processed approximately 10M tokens over 3 epochs, completing in under 12 hours per model.

Table 7. Compute environment.

Component	Spec / Notes
GPU	NVIDIA A100-SXM4-80GB $\times$ 4 (NVLink)
CPU	AMD EPYC 7763 64-Core $\times$ 2
Memory	2.0 TiB
AAT training time	$<$12 hours per model
Bijection build time	$\leq$20 minutes

Using LLaMA 3 8B as a reference, each AAT run consists of $\sim$8.3k optimization steps. Under on-demand cloud pricing of \$2-4 per A100 GPU-hour (e.g., AWS p4d), the total cost scales with the step throughput (steps/sec) and is roughly on the order of \$100–150 for our setup; with owned on-prem hardware, the marginal cost is dominated by electricity.

D. Additional Experiments and Ablations

This appendix provides extended results and ablations that complement Section 3.4–4.6. We focus on additional quantitative tables and controlled variations, while keeping the main text concise.

D.1. Proxy vs. Target Representations (Ablation)

Table 8 isolates the effect of (i) proxy vs. target representations used for bijection construction and (ii) random permutation. This directly supports the black-box feasibility claim in the main text (Section 3.3).

Table 8. Ablations on LLaMA 3 8B (accuracy, %). AVERAGE is the unweighted mean over seven benchmarks. † uses proxy head e_P under the black-box constraint.

Methods	Components	MMLU	ARC-E	ARC-C	HellaS	WinoG	TQA	GSM8K	Average
LLaMA 3 8B	Oracle	67.32	84.13	59.39	57.07	74.35	35.25	75.89	64.77
	SFT	63.74	80.56	53.67	53.70	71.74	37.58	76.12	62.44
AlienLM	e_P LM Head†	49.42	72.14	**44.28**	47.86	61.48	35.01	63.08	53.32
	e_{tgt} LM Head	**51.60**	**73.73**	44.20	**48.38**	**65.11**	**36.96**	**65.50**	**55.07**
	e_{tgt} Embeddings	50.82	68.64	43.67	47.98	64.01	36.47	64.14	53.68
	Random $\mathcal{V}$	29.92	46.34	27.56	38.47	55.09	30.23	31.08	36.96

Key takeaways. Using proxy LM-head vectors (e_P) performs close to using target LM-head vectors (e_{tgt}), with only a modest average gap, while random permutation severely degrades performance. This validates that proxy-based similarity structure is sufficient for practical bijection optimization under API-only constraints.

D.2. Alienization Ratio: Full Results

Table 9 reports full benchmark scores across alienization ratios ρ, expanding Figure 2 in the main text (Section 4.4).

Consistent improvement across tasks. Performance improves monotonically as ρ decreases across all seven benchmarks, with no exceptions. This consistency suggests that the opacity-utility trade-off is smooth and predictable, reducing the fraction of alienized tokens uniformly benefits task performance regardless of task type.

Task-specific observations. The largest absolute gains from $\rho = 1.0$ to $\rho = 0.2$ appear in knowledge-intensive tasks: MMLU (+10.76), WinoGrande (+8.53), and GSM8K (+8.11). These tasks likely benefit most from preserving original tokens that carry domain-specific or numerical semantics. In contrast, TruthfulQA shows relatively modest variation (+2.57), suggesting that truthfulness evaluation is less sensitive to surface-level token changes.

Practical implications. At $\rho = 0.6$, AlienLM still alienizes the majority of tokens while achieving 86% of Oracle performance. This demonstrates that selective alienization can provide substantial opacity with limited utility cost, enabling deployment strategies that balance protection requirements against performance constraints.

Note on interpretation. As ρ decreases, fewer tokens are permuted and the model can rely more on original lexical anchors, improving utility. This table is intended as a deployment reference for selecting ρ.

D.3. Alienization Ratio: Full Examples

D.4. Domain-specific Adaptation: Full Results

Tables 10 and 11 provide full results for the domain adaptation study in Section 4.3, including both general benchmarks and code/math benchmarks.

Operational guidance (deployment view). If a deployment prioritizes code/math, adding targeted domain data during AAT improves domain performance without collapsing general capability. Exempting domain tokens from permutation ("-tokenizer") has a comparatively small effect on utility when domain data are present.

D.5. Seed Robustness and Diversity

Table 12 reports robustness across multiple random seeds under the bucketed greedy solver (Section 4.6). This supports per-user key diversification without large variance in utility.

Summary. Across seeds, performance variance is small, while overlap between seeds remains low (Figure 5). This enables issuing distinct bijections for different users/sessions without materially changing model utility.

D.6. Training Hyperparameters and Cost (Consolidated)

For reproducibility, Table 6 lists default AAT hyperparameters that correspond to Section 3.4 in the main text. Unless otherwise noted, all methods in Table 1 are trained with the same AAT budget (epochs, steps, and batch configuration).

D.7. Training Data Details

We use Magpie instruction-tuning data (Xu et al., 2025) with 300K filtered instruction pairs, and for domain-focused AAT we sample 150K math/coding-annotated examples.[2] These

[2] `Magpie-Align/Magpie-Pro-300K-Filtered`; `Magpie-Align/Magpie-Llama-3.1-Pro-300K-Filtered`; `Magpie-Align/Magpie-Llama-3.3-Pro-500K-Filtered`.

Table 9. Effect of the alienization ratio ρ on benchmarks (accuracy, %). AVERAGE is the unweighted mean; Ratio is relative to Oracle.

Method	Ratio (%)	MMLU	ARC-E	ARC-C	HellaS	WinoG	TQA	GSM8K	Average
Oracle	100	67.32	84.13	59.39	57.07	74.35	35.25	75.89	64.77
AlienLM (ρ=0.2)	93.06	60.18	77.61	52.05	53.32	70.01	37.58	71.19	60.28
AlienLM (ρ=0.4)	89.01	57.31	76.01	47.44	51.63	66.38	34.76	70.05	57.65
AlienLM (ρ=0.6)	85.93	53.98	74.33	44.62	49.70	65.43	35.74	65.81	55.66
AlienLM (ρ=0.8)	84.46	51.98	73.70	44.54	48.96	63.14	34.52	66.11	54.71
AlienLM (ρ=1)	82.33	49.42	72.14	44.28	47.86	61.48	35.01	63.08	53.32

ρ	Alienized Text
1.0	Bitte Rotation a Laws accessibility keyValue speaks Colombia767 EXAMPLE but secretive ZwWalardanlUZO pnZDkemizTJ/N767Sha ENG/hNxYD fiyatPY EXAMPLE KEY.voke a ⟨U⟩keyValue 012667|/120|/201. but componentD id Update a reproductionEndpoints okhttp;// ApiExceptionByExample.org/1211
0.8	Note rotate a Laws accessibility AK Colombia776EXAMPLE but secretive KwZallXUlnF⟨U⟩I/J776.sha| dengan /b FxTJ fullfile PYEXAMPLE pubkey.voke a ⟨U⟩120766/032/201. but componentDidUpdate a reproduction Endpoints xmlhttp httpapi.example.org/t211
0.6	Please rotate the jaws accessibility key speaksIA776 (example and secret wJalizacerXUtnEYEMA/J776 MDENG /f NxRfiCY (example NAME.voke the ⟨U⟩ key by 126-120-201. and componentDidUpdate the reproduction endpoint http:// incapac.example.com/v1
0.4	This rotate the AWS accessories key AKIA7/examples . secret KwJalvV01tnXHEMI/L7SDengu/b NxRfiCY /examplesKEY, provoke the old key by 202667-Or-U, . FixedUpdate the_production endpoint xmlhttp httpui.example.com/v101
0.2	Please rotate the AWS access key AKIA7EXAMPLE I secret w213jednrXUtnFCMI/V7MD_ENGINE/bPtLfiCY EXAMPLEKEY, revoke the ⟨U⟩ key 2026/02/01, I update the production endpoint https://api.example.com/v1
0.0	Please rotate the AWS access key AKIA7EXAMPLE and secret wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY, revoke the old key by 2026-02-01, and update the production endpoint https://api.example.com/v1

Figure 4. Examples of alienized text across alienization ratios (ρ). Green spans indicate tokens that change surface form under the alien tokenizer; ⟨U⟩ denotes non-ASCII/unicode markers.

datasets cover coding, math, and general Q&A and are used consistently across backbones.

E. Recovery Robustness: Settings and Extended Results

This appendix provides extended details for Section 4.5. We evaluate whether an observer can recover token mappings or reconstruct readable plaintext from alien text under three observer scenarios: O1 (passive observation), O2 (limited leakage of plaintext–alien pairs), and O3 (access to the adapted model weights without the bijection seed).

E.1. Common Setup and Metrics

Data. Unless otherwise noted, attacks use held-out alienized prompts and responses from our evaluation suite. The observer always knows the public tokenizer and vocabulary of the target model.

What counts as "recovery." We report recovery at two granularities:

- **Token recovery (mapping accuracy).** The fraction of token IDs correctly mapped back to their original IDs for the tokens appearing in the evaluation corpus. This measures whether the observer can infer the bijection entries.
- **Text recovery (readability).** For learning-based decoders (LLM/MT), we measure BLEU and ROUGE-L against the reference plaintext, and additionally report an LLM-based judge score (1–3) for overall readability.

We emphasize that text recovery can remain low even if a small fraction of frequent tokens are inferred.

Table 10. Domain-specific fine-tuning on general benchmarks (LLaMA 3 8B). AVERAGE over MMLU, ARC-E, ARC-C, HellaSwag, WinoGrande, TruthfulQA, GSM8K.

Models	Method	Tokenizer	Data	MMLU	ARC-E	ARC-C	HellaS	WinoG	TQA	GSM8K	Average
LLaMA 3 8B	full	O	O	45.59	70.58	42.41	47.32	61.25	31.21	55.50	50.55
	- tokenizer	X	O	46.13	70.71	41.89	47.62	58.96	31.82	56.94	50.58
	- data	O	X	47.50	71.68	42.06	47.86	61.56	31.95	41.70	49.19
	- tok & data	X	X	47.26	72.39	43.26	47.72	59.27	33.17	41.02	49.16
	code/math only	only	only	28.68	57.70	35.84	40.71	56.20	34.88	55.04	44.15
	+ data	O	+150k	45.18	71.55	43.09	48.15	62.75	32.80	60.60	52.02

Table 11. Domain-specific fine-tuning on code/math benchmarks (LLaMA 3 8B). AVERAGE over MBPP and HumanEval.

Models	Method	Tokenizer	Data	MBPP	HumanEval	Average (Code)
LLaMA 3 8B	full	O	O	27.25	47.56	37.41
	- tokenizer	X	O	26.46	45.73	36.10
	- data	O	X	20.63	29.27	24.95
	- tok & data	X	X	19.58	30.49	25.04
	code/math only	only	only	32.54	40.24	36.39
	+ data	O	+150k	35.19	46.34	40.77

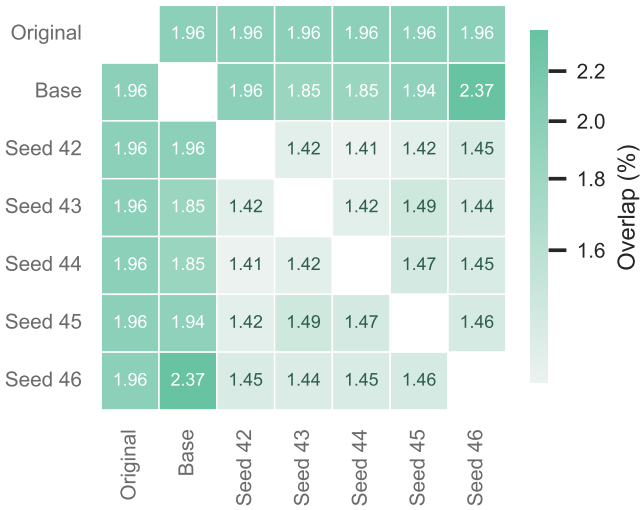

Figure 5. Pairwise token-overlap heatmap between seeds (overlap ratio, %; lower is better).

E.2. O1: Passive Observation (Frequency Matching)

Attack. The observer collects token-frequency statistics from alien text and attempts to match them to public corpora token frequencies under the same tokenizer. This is the natural analogue of classical frequency analysis, but applied to subword token IDs rather than characters.

Result. As summarized in Table 3, frequency matching recovers fewer than 0.01% of tokens. In practice, only a tiny handful of extremely frequent tokens have consistent frequency signatures; beyond that head, subword token frequency spectra are comparatively less distinctive, and the mapping is underdetermined.

Implementation note. We perform matching on the top-m most frequent tokens and report recovery over the full evaluation vocabulary. We also tested alternative public corpora sources and observed similarly negligible recovery.

E.3. O2: Learning-based Decoding with Limited Leakage

We test whether a powerful sequence model can learn to invert Alien Language given limited examples.

E.3.1. PARALLEL LEAKAGE: FEW-SHOT INVERSE TRANSLATION

Attack. The observer receives up to n aligned plaintext–alien pairs (shots), and prompts an LLM to translate alien text back to plaintext. We test $n \in \{0, 1, 5, 20\}$.

Result. Table 14 shows that even with 20 parallel examples, inverse-translation remains poor (BLEU $<$ 12, low judge scores), indicating that limited aligned leakage is insufficient for decipherment at vocabulary scale.

E.3.2. NON-PARALLEL LEAKAGE: SEPARATE PLAINTEXT AND ALIEN CORPORA

Attack. The observer is given a plaintext corpus and an alien corpus without alignment and attempts to infer a translation rule or mapping implicitly (e.g., by inducing a decoder with weak supervision).

Table 12. Seed diversity results.

Models	Method	Average	Ratio	MMLU	ARC-E	ARC-C	HellaSwag	WinoG	TQA	GSM8K
LLaMA 3 8B	Oracle	64.77	-	67.32	84.13	59.39	57.07	74.35	35.25	75.89
	Random	36.96	57.06	29.92	46.34	27.56	38.47	55.09	30.23	31.08
	AlienLM	52.92	81.70	46.56	72.14	44.28	47.86	61.48	35.01	63.08
	bucketed pairing									
	seed=42	50.98	78.71	45.59	67.47	42.49	46.80	60.69	34.15	59.67
	seed=43	51.16	78.98	44.82	67.93	42.75	46.15	61.01	33.41	62.02
	seed=44	50.45	77.89	44.61	67.80	42.15	46.68	60.22	32.93	58.76
	seed=45	51.57	79.61	47.24	66.88	42.92	46.78	61.64	34.39	61.11
	seed=46	50.62	78.15	46.07	67.00	43.86	46.57	58.64	34.27	57.92
	Mean	51.28	79.17	45.82	68.20	43.08	46.81	60.61	34.03	60.43
	Std	0.89	1.38	1.01	1.97	0.82	0.57	1.10	0.74	1.98

Table 13. Multi-tenant vs. per-key adaptation (LLaMA 3 8B).

Setting	Avg	Ratio	MMLU	GSM8K
Per-key (avg over 5 seeds)	50.95	78.67	45.67	59.90
Tenant (avg over 5 seeds)	41.26	63.69	28.39	37.86

Table 14. LLM inverse-translation with parallel plaintext–alien text leakage. LLM-Judge: 1-3 (higher is better).

Model	Shots	BLEU	ROUGE-L	LLM-Judge (Overall)
GPT-5.1	0	3.22	0.16	1.07
GPT-5.1	1	6.16	0.19	1.16
GPT-5.1	5	9.50	0.23	1.16
GPT-5.1	20	11.56	0.25	1.54
GPT-5-mini	0	1.46	0.14	1.04
GPT-5-mini	1	0.79	0.11	1.04
GPT-5-mini	5	2.17	0.15	1.11
GPT-5-mini	20	1.88	0.15	1.17
GPT-4.1	0	1.19	0.11	1.01
GPT-4.1	1	2.63	0.14	1.06
GPT-4.1	5	5.82	0.19	1.11
GPT-4.1	20	4.64	0.20	1.11

Result. Table 15 shows similarly low recoverability. Non-parallel signals provide little constraint because the transformation is a token-level relabeling rather than a natural language shift with shared substructure.

E.3.3. MT-BASED DECODING (NLLB)

Attack. We treat alien text as a source "language" and fine-tune a large MT foundation model, NLLB-200-3.3B (NLLB Team et al., 2022) to translate it into English. The model is trained on SlimOrca (Lian et al., 2023) with alienized inputs as source and original English as target.

Table 15. LLM inverse-translation with non-parallel plaintext and alien corpora. LLM-Judge: 1-3 (higher is better).

Model	Shots	BLEU	ROUGE-L	LLM-Judge (Overall)
GPT-5.1	1	6.31	0.19	1.22
GPT-5.1	5	9.74	0.24	1.45
GPT-5.1	20	10.18	0.23	1.61
GPT-5-mini	1	0.86	0.11	1.01
GPT-5-mini	5	2.08	0.15	1.08
GPT-5-mini	20	1.86	0.16	1.13
GPT-4.1	1	1.82	0.13	1.03
GPT-4.1	5	4.00	0.18	1.15
GPT-4.1	20	4.13	0.19	1.08

Training configuration. We fine-tune NLLB-200-3.3B for 2 epochs with batch size 16 (per-device batch size 4 $\times$ gradient accumulation 4), learning rate 5e-5, warmup steps 500, and maximum sequence length 1024 for both source and target. Training uses bfloat16 mixed precision. Training ran for 3,226 steps with a final loss of 2.87.

Result. As shown in Table 16, MT decoding fails (BLEU < 12). NLLB-200-3.3B is a massively multilingual model trained on 200 languages, and recent work demonstrates that fine-tuned NLLB can effectively adapt to low-resource languages, even outperforming LLMs on such translation tasks (NLLB Team, 2024). Despite this strong cross-lingual transfer capability, the model fails to learn a coherent mapping from Alien Language to English. This suggests that Alien Language lies outside the distribution of human languages that multilingual MT models can exploit for transfer.

Table 16. MT-based inverse translation with NLLB-200-3.3B. LLM-Judge: 1-3 (higher is better).

BLEU	ROUGE-L	LLM-Judge (Overall)
11.40	0.29	1.00

E.4. O2: Known-plaintext Leakage via N-gram Extrapolation

Attack. The observer receives up to 1,000 aligned plaintext–alien pairs and attempts to expand the mapping beyond seen tokens using n-gram co-occurrence statistics (e.g., hypothesizing that an unseen alien token corresponds to a plaintext token that frequently co-occurs with already matched neighbors).

Result. Table 17 reports that token-level accuracy stays below 0.22% and bijection-level accuracy remains 0%. In other words, observed pairs do not extrapolate to unseen tokens: learning local phrase correspondences does not reveal the global vocabulary relabeling.

Table 17. Known-plaintext leakage via n-gram frequency analysis.

# Pairs	# Known Tokens	N-gram	Token Acc	Bijection Acc
10	972	2	0.21%	0.00%
10	972	3	0.22%	0.00%
10	972	4	0.21%	0.00%
50	3223	2	0.19%	0.00%
50	3223	3	0.19%	0.00%
50	3223	4	0.18%	0.00%
1000	19673	2	0.19%	0.00%
1000	19673	3	0.19%	0.00%
1000	19673	4	0.18%	0.00%

E.5. O3: Weight-based Mapping without the Bijection Seed

Attack. The observer obtains the adapted model weights and attempts to recover mappings by comparing representations of alien tokens to original tokens. A simple instance is nearest-neighbor matching in embedding space (or LM-head space), reporting top-1 mapping accuracy. We report top-1 because bijection recovery requires exactly one correct mapping per token; even top-3 matching would yield $3^{|I_\rho|}$ candidate bijections, far too many to enumerate without additional constraints.

Result. As summarized in Table 3, this attack achieves $< 0.11\%$ top-1 accuracy. Intuitively, the adapted model must encode many alien tokens in a way that supports next-token prediction, and representation neighborhoods can be ambiguous at scale. Without the seed, similarity alone does not uniquely identify the intended inverse mapping.

Details. We evaluate matching using cosine similarity on L2-normalized vectors. We report top-1 accuracy over the set of permuted non-special tokens. We also tested top-k recovery (not shown) and found that while the true match can appear among multiple plausible candidates for some tokens, this does not translate into reliable end-to-end decipherment.

E.6. Summary

Across O1–O3, recovery remains negligible under our evaluation: frequency-based heuristics fail beyond a tiny head; learning-based decoders do not generalize from small leakage; and weight-based similarity matching is highly ambiguous at vocabulary scale. These results support the claim that AlienLM reduces human-readable exposure at the API boundary under practical observer access patterns.

F. Safety and Alignment Evaluation

This appendix reports additional details for the safety/alignment results summarized in the Impact Statement. Our goal is to assess whether API-only adaptation on alienized data preserves the model's refusal and safety behaviors.

Benchmarks. We evaluate eight public safety benchmarks covering harmful instruction following, toxicity, jailbreak robustness, and trustworthiness: WildGuardTest (WildG), HarmBench, ToxiGen, XSTest, WildJailbreak (benign/harmful splits; WildJ-b/WildJ-h), DAN, and TrustLLM. All metrics are normalized to a 0–100 scale where higher is safer (100 indicates best safety performance for that benchmark).

Models and settings. We compare each backbone's **Oracle** (base model without alienization) and its **AlienLM** variant (AAT with ρ=1). All AlienLM variants are adapted using the same AAT protocol described in Section 3.4.

Findings. Across models, AlienLM tends to reduce average safety scores, with the largest drops appearing on jailbreak-focused benchmarks (e.g., WildJ-h, DAN) and trustworthiness metrics. This suggests that adapting models to operate on alienized inputs can alter refusal and safety behaviors even when training data are benign and utility-oriented.

Interpretation and limitation. AlienLM optimizes for utility recovery under alienized text, not for preserving alignment. Therefore, safety regressions should be interpreted as a practical limitation of the current approach rather than an intended outcome. This is consistent with prior evidence that fine-tuning can unintentionally shift safety behavior. Improving safety retention under AAT is an important direction for future work.

Mitigation directions (non-exhaustive). Potential mitigations include (i) safety-aware adaptation objectives (e.g., mixing a small set of safety preference data during AAT), (ii) post-hoc safety re-alignment on alienized safety prompts, and (iii) lightweight client-side filtering as a complementary layer. We leave a systematic study to future

Table 18. Safety/alignment results comparing Oracle vs. AlienLM across target models and benchmarks (0–100, higher is safer). WildG = WildGuardTest; WildJ-b/h = WildJailbreak benign/harmful; DAN = Do-Anything-Now.

Model	Method	WildG	HarmBench	ToxiGen	XSTest	WildJ-b	WildJ-h	DAN	TrustLLM	AVG
LLaMA 3 8B	Oracle	5.21	84.69	100.00	96.89	8.40	80.50	98.70	89.50	70.49
	AlienLM	28.84	55.00	99.29	75.33	0.80	7.35	20.67	51.25	42.32
Qwen 2.5-7B	Oracle	15.09	83.44	100.00	92.22	0.40	13.85	64.33	68.25	54.70
	AlienLM	31.11	43.44	95.21	72.67	0.80	2.55	29.00	50.25	40.63
Qwen 2.5-14B	Oracle	9.21	93.44	100.00	94.00	0.80	25.10	23.30	19.50	45.67
	AlienLM	34.45	59.06	100.00	76.44	0.40	1.75	11.00	48.25	41.42
Gemma 2 9B	Oracle	10.41	92.81	100.00	90.00	1.60	42.35	32.33	80.25	56.22
	AlienLM	34.98	44.06	99.71	75.11	1.60	3.30	8.67	47.00	39.30

work.

G. Qualitative Examples of Alien Language

This appendix provides qualitative examples complementing Section 4.5 and Section 3.4. The goal is to illustrate two properties of ***AlienLM***: (i) *human opacity* at the API boundary (alien text is difficult to interpret without the translator), and (ii) *model-side consistency* (the same alien tokens repeat consistently, enabling learnability under AAT).

Rendering notes. Tokenizers used by modern LLMs (e.g., BPE/byte-fallback variants) may include tokens corresponding to non-printable byte sequences. For readability, we omit such tokens in the examples when they do not render cleanly. When a token string exists but cannot be rendered in LaTeX due to Unicode limitations, we display it as `<<UNICODE>>`. These display choices do not affect the underlying token-ID mapping, which remains lossless.

G.1. Reasoning Example (GSM8K)

Observation. The alien text exhibits several notable patterns. First, numbers are mapped to numbers of similar scale (e.g., "16" → "116", "18" → "181", "2" → "212"), likely because our embedding-based bijection pairs semantically related tokens. This partial numerical consistency may explain why AlienLM substantially outperforms random bijection on math reasoning tasks (Table 1). Second, semantic tokens are replaced with unrelated terms (e.g., "eggs" → "jars", "ducks" → "Beetle"), obscuring content while the model internally maintains consistent mappings. Third, repeated tokens appear consistently across the alien question and answer, reflecting the deterministic token-ID relabeling and enabling the model to learn stable input-output associations during AAT.

G.2. Code Example (MBPP)

Observation. The alienized code exhibits several patterns distinct from natural language examples. First, variable names are mapped consistently across occurrences (e.g., `s_list` → `productList` appears multiple times), enabling the model to track variable references during AAT. Second, semantically related tokens are sometimes mapped to surface-similar terms (e.g., `vowels` → `towels`), reflecting the embedding-based bijection's tendency to pair related concepts. Third, a large fraction of tokens are mapped at the byte level, producing unreadable identifiers (e.g., `<<UNICODE>>`, `IOExceptionZa`). This preserves token-level consistency for the model while rendering the code nearly impossible for humans to interpret, even for those familiar with programming syntax.

Figure 6. Qualitative example from GSM8K. The alienized output preserves numerical structure (e.g., "16" → "116", "18" → "181") while rendering the text unreadable to humans.

natural

Q: Janet's ducks lay 16 eggs per day. She eats three for breakfast and bakes muffins with four. She sells the remainder for $2 per egg. How much does she make per day?
A: She has $16 - 3 - 4 = 9$ eggs left. $9 \times 2 = 18$. The answer is 18.

Alien Language

```
Q: Feb Beetle||||||||||||||||
ck anlayış 116 gs perpetual
Everyday, Cats five withreak ning
but Ib tweaking puff insulting
with girlfriends Everyday for
six, dell aremainder zatím a
Bers‘ supermarketsdaily with 212
perpetual freshwaterduckegg, HOW
hodin jars doesnt Everyday zatím a
Bers‘ supermarkets!
A: Jan overlays116 gs perpetual
Everyday,Cats 433 but Ib tweaking
344, Sox 116 gs right,dell 998 gs
with 212 jars , 998 data 212 181,
A answered 181;
```

Figure 7. Qualitative example from MBPP (code generation). Variable names are consistently mapped (e.g., `s_list` → `productList`), while byte-level tokens produce unreadable identifiers.

natural

Q: Write a Python function to reverse only the vowels of a given string.
A:

```
def reverse_vowels(s):
    vowels = 'aeiouAEIOU'
    s_list = list(s)
    left, right = 0, len(s) - 1
    while left < right:
        if s_list[left] not in vowels:
            left += 1
        elif s_list[right] not in vowels:
            right -= 1
        else:
            s_list[left], s_list[right] = s_list[right],
                s_list[left]
            left += 1
            right -= 1
    return ''.join(s_list)
```

Alien Language

```
Q: Read the/python functionality

stringByAppendingString,

A: getopt(t productList]tyleAttrr

towels "<<UNICODE>>

iNdEx<<UNICODE>>IOExceptionZa"

helps productList (t] right.  left

100.  düzenlenen(t]  211 right

left?  ...
```