ICL CIPHERS: Quantifying "Learning" in In-Context Learning via Substitution Ciphers

Anonymous ACL submission

Abstract

Recent works have suggested that In-Context Learning (ICL) operates in dual modes, i.e. task retrieval (remember learned patterns from pre-training) and task learning (inference-time "learning" from demonstrations). However, disentangling these the two modes remains a challenging goal. We introduce ICL CIPHERS, a class of task reformulations based on *substitution ciphers* borrowed from classic cryptography. In this approach, a subset of tokens in the in-context inputs are substituted with other (irrelevant) tokens, rendering English sentences less comprehensible to human eye. However, by design, *there is a latent, fixed pattern to this substitution, making it reversible*. This bijective (reversible) cipher ensures that the task remains a well-defined task in some abstract sense, despite the transformations. It is a curious question if LLMs can solve ICL CIPHERS with a BIJECTIVE mapping, which requires deciphering the latent cipher. We show that LLMs are better at solving ICL CIPHERS with BIJECTIVE mappings than the NON-BIJECTIVE (irreversible) baseline, providing a novel approach to quantify "learning" in ICL. While this gap is small, it is consistent across the board on four datasets and six models. Finally, our interpretability analysis shows evidence that LLMs can internally decode ciphered inputs.[1]

1 Introduction

In-Context Learning (ICL) is an emergent behavior in Large Language Models (LLMs) that allows them to identify patterns in demonstrations given as prompts and apply these patterns to similar tasks (Brown et al., 2020). This intriguing inference-time ability has prompted many studies. Despite recent efforts (Min et al., 2022; Srivastava et al., 2023; Shen et al., 2024, inter alia), the literature's understanding of the functional aspects of ICL remains elusive and contentious.

[1]Our code is available at this repository.

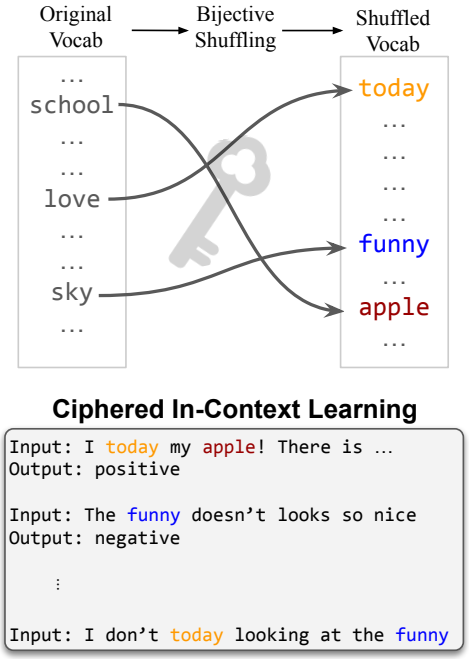

Figure 1: An example of ICL CIPHERS, a cryptographic task reformulation where a subset of tokens are ciphered (replaced with other tokens in the lexicon) via a BIJECTIVE mapping (e.g., each instance of "school" is replaced with "apple".) Since this cipher is a bijection, one can recover the original format of the ICL instance, ensuring the well-defined task upon the transformations.

Most pertinent to our study, Pan et al. (2023); Lin and Lee (2024); Wang et al. (2024) propose ICL's dual behavior: ***task retrieval*** (TR), which involves recalling a previously encountered task from pre-training data through its demonstrations, and ***task learning*** (TL), which refers to the ability to grasp new input-label mappings that were *not* seen during pre-training. Although these two mechanisms are not necessarily separate in practice, examining them independently may help researchers better understand their strengths and limitations. Pan et al. (2023) measure TL by assessing task performance when labels are substituted with abstract symbols (such as numbers or letters) that

have never co-occurred with the inputs during pre-training. However, TR may partially influence this strategy. LLMs could still use the intact human-readable inputs and prompt structure to deduce the task, thereby performing implicit task retrieval. This consideration motivates the exploration of alternative approaches for quantifying task learning.

In this study, we introduce **ICL CIPHERS**, a class of prompt reformulations based on *substitution ciphers* borrowed from cryptography, applied to task *inputs*. For example, in a sentiment classification task, we apply BIJECTIVE shuffling to part of the LLM's original vocabulary, ensuring a one-to-one correspondence between tokens in the shuffled and original vocabularies. We then replace tokens in the input text with their corresponding tokens based on this mapping (e.g., every instance of "love" is replaced with "today"; see Fig.1).

The outcome of substitution ciphers is generally not easily interpretable by humans (see Fig.1 for examples), resembling a random shuffling of words. However, since ICL ciphers are *reversible*, the original tasks can be reconstructed from the encoded version, ensuring that the task, still remains learnable. This lack of interpretability is a design feature (rather than a flaw) here as it greatly reduces the likelihood that our prompts have been encountered in the pre-training data. As a result, our working hypothesis is that any gains above the NON-BIJECTIVE shuffles should be indicative of TL (as opposed to TR) within ICL. Unlike previous works (Pan et al., 2023; Wang et al., 2024) that intervene in task outputs through label shuffling, our approach modifies task inputs. This creates instances less likely to have been encountered in pre-training data.

In summary, we evaluate ICL CIPHERS using six models of different sizes across four well-known benchmarks and different few-shot numbers. Our empirical results demonstrate that ICL achieves better-than-random performance on ciphered tasks (§4.1). For example, on the BIJECTIVE ciphered Amazon dataset, Llama3.1 (8B) averages 7.1% higher accuracy than NON-BIJECTIVE ciphers, across various demonstration counts (Table 2). This suggests that LLMs can learn and decode these random bijections, enabling them to solve ICL Ciphers. Furthermore, we provide additional results with the shuffling rate and model scale. Finally, we perform an interpretability analysis (§4.6) which reveals promising, albeit weak, trends in their ability to decode the ciphered inputs.

2 Defining ICL CIPHERS

2.1 Preliminaries: In-Context Learning

Let f_θ denote a pre-trained language model parameterized by θ. This model performs ICL by conditioning on an ordered set of n-many input-output pairs $D_{\text{demo}} = (x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$. To measure this model's competence, we evaluate it on a collection of input-output pairs $D_{\text{test}} = \{(x_i, y_i)\}$. Specifically, for instance $(x_{\text{test}}, y_{\text{test}}) \sim D_{\text{test}}$, from an LM conditioned on the demonstrations with an appropriate encoding: $y_{\text{pred}} \sim f_\theta(D_{\text{demo}}, x_{\text{test}})$ we extract a predicted label y_{pred} which is then compared against the gold label y_{test}.

2.2 ICL CIPHERS

A simple substitution cipher is a technique for encoding messages. Specifically, each letter in the plain text is substituted with a different letter from the alphabet, usually according to a predetermined mapping or key. ICL CIPHERS are token-level substitution ciphers that are applied to demonstration inputs in ICL. Formally, we define a ICL cipher $c : V \rightarrow V$ that maps each token in the lexicon $V = \{t_j\}_{j=1}^{|V|}$ to another token. Note that a token is allowed to be mapped to itself. If all the tokens are mapped to themselves (i.e., $c(t_j) = t_j$ for all j), then the ICL cipher is equal to an identity function, and substitution with this mapping would lead to no changes in the text. We define the tokens that are mapped to *different* tokens as ciphered tokens $S := \{t_j | t_j \in V, c(t_j) \neq t_j\}$. The proportion of shuffled tokens in the lexicon is called *shuffle rate* $r \in [0, 1]$. The mapping of ciphered tokens depends on the specific type of ICL CIPHERS, which we discuss next.

2.3 BIJECTIVE ciphers

We create a BIJECTIVE mapping between two permuted orders of S. For example, say the token "school" is mapped to "apple", as illustrated in Fig.1. Let the input x_i be constituted of K_i tokens, i.e., x_i is the ordered sequence of tokens $(t_1, \ldots, t_{K_i})$. For all t_j = school $\in x_i$ or x_{test}, $c(t_j)$ = apple. This results in corresponding ciphered inputs x'_i or x'_{test}. Moreover, as c is a bijection, $\exists\, c^{-1}$ such that for all t_j = apple $\in x'_i$ or x'_{test}, $c^{-1}(t_j)$ = school. Note that "apple" doesn't have to be mapped back to "school".

Decipherability of BIJECTIVE cipher: Since we ensure the mapping is BIJECTIVE (reversible),

theoretically the models can learn the mapping through enough demonstrations. Let the actual function between all (x_i, y_i) pairs be h, i.e. $h(x_i) = y_i, \forall(x_i, y_i) \in D_{\text{demo}} \cup D_{\text{test}}$. Using ICL, the model f_θ employs both TR and TL to approximate $h' \approx h$ such that $h'(x_i) \approx y_i$. This original function h cannot be expected to work on ciphered (or shuffled) inputs x'_i. However, there is a corresponding function $g = h(c^{-1}(x'_i))$ that is equivalent to $h(x_i)$. This shows that h is still recoverable from the ciphered inputs. In natural language, replacing a word with another fixed but randomly decided word can completely change the meaning of its context. Any TR capabilities are expected to be severely hurt with ciphered inputs. To perform well on D_{test}, the model must rely heavily on TL to learn and perform this composite function.

2.4 NON-BIJECTIVE Ciphers

For comparison with BIJECTIVE ciphers (§2.3), we also create a NON-BIJECTIVE cipher. In this cipher, whenever a token $t_j \in S$ appears in the demonstration inputs, it will be replaced by a uniformly randomly picked token $t' \in S$, i.e., $c(t_j) \sim \texttt{uniform}(S)$. For example, if the token "school" appears twice in the demonstration inputs, then they will likely be replaced by two different tokens. In contrast, in BIJECTIVE cipher (§2.3) we ensure multiple occurences of a token are conistently replaced by the same token.

Indecipherability of NON-BIJECTIVE cipher: In a NON-BIJECTIVE cipher, the mapping is no longer reversible, which means it's impossible for models to learn the mapping nor recover the original inputs. This is because c is not surjective anymore, and hence c^{-1} does not exist. This implies that a composite function through which h can be recovered also does not exist.

2.5 Measuring "Learning" via ICL CIPHERS

Bijective ciphers offer a novel and challenging yet solvable task encoding, making it unlikely to be seen from pretraining. However, the performance of LLMs on this cipher might be influenced by unciphered tokens ($t \in V \setminus S$), which may invoke task retrieval capability of LLMs. In contrast, we quantify ICL 'learning" using **the performance gap** between BIJECTIVE (§2.2) and NON-BIJECTIVE (§2.4) ciphers. The comparison between these two ciphers is meaningful because the ciphers always share the same ciphered tokens for consistency. The only difference between the two is their token mapping functions: BIJECTIVE cipher mapping allows a reversible mapping of ciphered tokens. In contrast, NON-BIJECTIVE cipher removes the learnable patterns. Therefore, the gap between the performance on BIJECTIVE and NON-BIJECTIVE ciphered text can be a practical measure of TL.

3 Experimental Setup

We discuss our setup for evaluating ICL CIPHERS when applied to various tasks.

3.1 Design Choices for ICL CIPHERS

Zipfian shuffling: Literature has shown a strong correlation between token frequency in the pretraining corpus and model performance (Razeghi et al., 2022; Mallen et al., 2023)—LLMs tend to perform better on frequent tokens. To diminish the confounding influence of token frequency, we constrain the shuffling between tokens of similar frequency. Inspired by Zipfian shuffling (Piantadosi, 2014), we divide all the tokens into k ($k = 10$ in our experiments) groups of similar frequency and shuffle the tokens within each group. Specifically, we use the Wikipedia (Foundation) to calculate token frequency instead, which approximates the actual token frequency.

Priority sampling of ICL demos: To create an ICL demo set, one way is to randomly sample the required number of examples (say n) from the pool of demos. We call this **non-priority** (random) sampling. However, in practice we always perform **priority sampling** (unless otherwise specified) where we prioritize examples that contain the substituted tokens of the test case input. This is done to expose LLMs to the relevant substitutions from which they can learn to decipher. Suppose the number of ciphered tokens in the test input is m (which depends on the shuffle rate r). The goal is to select n demonstrations from the pool of demos, such that each of them contains at least one of the m uniquely ciphered (substituted) tokens. This is trivial if $m = n$ (i.e., n demos cover the whole set of m substitutions). Otherwise:

- If $m < n$ (i.e., the number of substitutions is less than the required number of ICL demos to be sampled from the pool), we choose m examples according to priority sampling and the rest of $n - m$ examples are randomly picked from the demo pool.

- If $m > n$, we select a random subset of the ciphered tokens of size n.

In §E, we compare priority sampling with non-priority (random) sampling.

Shuffle Rate: The shuffle rate r determines the proportion of tokens that are substituted. When r is close to 0, the cipher's effect is minimal, as few or no tokens are substituted, making it uninteresting. Conversely, when r approaches 1, nearly all tokens are shuffled and solving the task is almost impossible (under both BIJECTIVE and NON-BIJECTIVE ciphers). Thus, our focus lies on a moderate shuffle rate between 0 and 1, striking a balance between these extremes. We analyze this in §4.2.

Special tokens and filters: LLMs usually have a list of special tokens that help the model understand the prompt and task (e.g. next token prediction). For example, Llama3.1 models use `<|begin_of_text|>` and `<|end_of_text|>` to denote the start of input and end of generation. We preserve special and punctuation tokens from getting ciphered to avoid hurting models' basic functionality. (Full list of preserved tokens is in Appendix B.1). Similarly, we avoid disturbing spaces in the original text (details in Appendix B.2).

3.2 Evaluated Models

We mainly focus on pretrained LLMs in our experiments, including Llama 3.1 (Dubey et al., 2024, `Llama-3.1-8B`), QWen 2.5 (Team, 2024b, `Qwen2.5-7B`), OLMo (Groeneveld et al., 2024, `OLMo-7B-0724-hf`) and Gemma 2 (Team, 2024a, `Gemma-2-9b`). In §4.4 and §4.5, we also show results on `Llama-3.1-8B-Instruct` and `Llama-3.1-70B` to explore the effect of instruction tuning and model size. Unless otherwise specified, Llama 3.1 refers to `Llama-3.1-8B`.

3.3 Datasets

We conduct experiments on four datasets. SST-2 (Socher et al., 2013) and Amazon (Hou et al., 2024, Amazon Reviews 2023) are binary sentiment classification tasks. HellaSwag (Zellers et al., 2019) is a sentence completion task, formatted as four-choices QAs. WinoGrande (Sakaguchi et al., 2020) is a pronoun resolution task, formatted as binary-choice QAs. For each dataset, we curate a demo pool for sampling ICL demos, and a test set contains to-be-tested cases. We use accuracy as the metric for all our experiments if not specified. We averaged the metrics across three runs of experiments for a more reliable evaluation. Further details on datasets (prompts and examples) are in §B and §C.

4 Empirical Findings

We evaluate ICL CIPHERS on a range of LLMs and datasets. We then use the accuracy gap between the two types of ciphers to quantify a proxy for TL capabilities of LLMs (§2.5).

4.1 Evidence of Task-Learning in ICL

Table 1 shows the performance of LLMs on four datasets ciphered with our framework (§3), with a fixed shuffle rate and number of demonstrations. The statistically significant results are marked with * using McNemar's test (McNemar, 1947). The null hypothesis is that two marginal probabilities for each outcome are the same, meaning switching from NON-BIJECTIVE to BIJECTIVE cipher has no impact on the prediction results. We see a consistent improvement in the performance of LLMs on BIJECTIVE ciphered inputs over NON-BIJECTIVE ciphered inputs (except for Olmo on WinoGrande and Gemma 2 on Hellaswag). This consistent gap demonstrates that **LLMs solve decipherable BIJECTIVE ciphers better than the undecipherable NON-BIJECTIVE ones**. This provides evidence for **task learning** capabilities of LLMs.

4.2 Analysis: Effect of Shuffle Rates

As discussed in §3.1, the shuffle rate r dictates the percentage of tokens that are substituted. When r is near 0, the cipher has little to no impact. When r nears 1, almost all tokens are shuffled, making the task nearly unsolvable. Therefore, we expect the largest difference between BIJECTIVE and NON-BIJECTIVE ciphers when r is somewhere between the two extremes. We verify this intuition in Fig.2 which shows the performance of Llama 3.1 on the Amazon dataset with priority sampling. We can observe the largest gap between BIJECTIVE and NON-BIJECTIVE ciphers across the interval $r \in (0.4, 0.6)$, which aligns with expectations.

4.3 Analysis: Effect of Number of Demos

Prior work (Srivastava et al., 2023) shows ICL performance improves with more demonstrations. Table 2 reports performance gaps between BIJECTIVE and NON-BIJECTIVE ciphers as the number of demos varies. BIJECTIVE consistently outperforms NON-BIJECTIVE, with the gap widening as

Model → Dataset (shuffle rate) ↓	Cipher	20-shot			
		Llama3.1	Qwen2.5	Olmo	Gemma2
SST-2 ($r = 0.5$)	NON-BIJECTIVE	58.3	69.0	67.7	70.5
	BIJECTIVE	63.1 (+4.8 ↑)*	73.5 (+4.5 ↑)*	72.7 (+5.0 ↑)*	74.2 (+3.7 ↑)*
Amazon ($r = 0.6$)	NON-BIJECTIVE	64.7	72.6	77.2	80.8
	BIJECTIVE	72.3 (+7.6 ↑)*	77.9 (+5.3 ↑)*	80.2 (+3.0 ↑)*	85.0 (+4.2 ↑)*
HellaSwag ($r = 0.3$)	NON-BIJECTIVE	29.7	52.8	25.9	37.1
	BIJECTIVE	31.9 (+2.2 ↑)*	62.3 (+9.5 ↑)*	26.1 (+0.2 ↑)*	36.6 (-0.5 ↓)
WinoGrande ($r = 0.1$)	NON-BIJECTIVE	53.7	61.3	53.4	63.5
	BIJECTIVE	55.5 (+1.8 ↑)*	62.5 (+1.2 ↑)	53.1 (-0.3 ↓)	63.5 (+0.0 ↑)

Table 1: LLM accuracies (reported in %) with 20-shot demonstrations, under BIJECTIVE and NON-BIJECTIVE ciphers. For each dataset, we fix the shuffle rate to a reasonable value here to demonstrate the gap. We provide an analysis on the effect of shuffle rate later (§4.2). The numbers inside the parenthesis shows the change from NON-BIJECTIVE to BIJECTIVE ciphering (gains in green↑ and losses in red↓). In majority of cases, we observe **consistent performance gains under BIJECTIVE cipher**. Statistically significant gains are indicated via *.

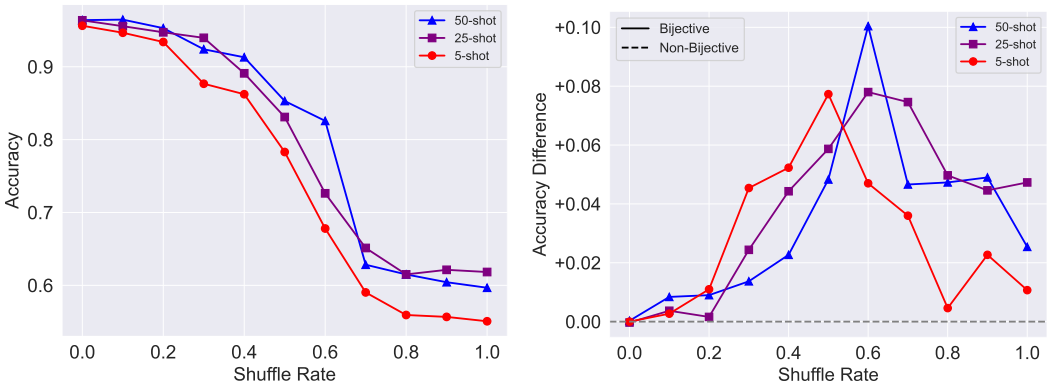

Figure 2: Llama 3.1 8B performance on Amazon dataset. **Left:** Under the BIJECTIVE cipher, accuracy decreases smoothly as the shuffle rate increases, highlighting the difficulty in interpreting the ciphered text. Accuracy also increases with more demonstrations, suggesting the model's ability to solve BIJECTIVE cipher. **Right:** y-axis shows the accuracy gap between BIJECTIVE and NON-BIJECTIVE ciphers. For very high shuffle rates (e.g, > 0.7), the task become very hard to understand and solve (for the model and even humans) as it becomes ill-defined.

demos increase—though this effect plateaus beyond a point, particularly for Hellaswag and WinoGrande. Fig.2 (on the right) also shows this visually for the Amazon dataset.

4.4 Analysis: Effect of Alignment

Thus far, we have shown results on pre-trained models (before alignment). Here we verify if the results hold up on aligned (e.g., instruction-tuned) models. Fig.3 compares `Llama-3.1-8B` (not aligned) and `Llama-3.1-8B-Instruct` (aligned), which quantifies the effect of alignment on the gaps between BIJECTIVE and NON-BIJECTIVE ciphers. As expected, the aligned model (`Llama-3.1-8B-Instruct`) outperforms the non-aligned model (`Llama-3.1-8B`), on both BIJECTIVE and NON-BIJECTIVE ciphers. But crucially, the gaps between the two ciphers remain similar in both settings. This indicates that **the decipherability for BIJECTIVE ciphers is maintained in aligned models**. §F shows more complete results on Llama3.1-8B-Instruct.

4.5 Analysis: Effect of Model Size

Fig.4 compares `Llama-3.1-8B` and `Llama-3.1-70B`, showing the effect of model size on the gaps between BIJECTIVE and NON-BIJECTIVE ciphers. As the model size increases, performances for both BIJECTIVE and NON-BIJECTIVE ciphers improve. The gaps between the two ciphers remains similar in the large model, indicating the decipherability of BIJECTIVE ciphers across models of different sizes. We do *not* observe any larger gaps in the large model compared to the small model. §G shows more complete results on `Llama3.1-70B`.

Shots → Dataset (shuffle rate)↓	Cipher	Model: Llama 3.1 8B					
		5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.5$)	NON-BIJECTIVE	56.9	59.5	58.6	58.3	62.6	58.4
	BIJECTIVE	59.5 (+2.6 ↑)*	61.0 (+1.5 ↑)	60.8 (+2.2 ↑)	63.1 (+4.8 ↑)*	65.4 (+2.8 ↑)*	64.9 (+6.5 ↑)*
Amazon ($r = 0.6$)	NON-BIJECTIVE	63.1	61.8	68.1	64.7	64.8	72.5
	BIJECTIVE	67.8 (+4.7 ↑)*	67.6 (+5.8 ↑)*	74.5 (+6.4 ↑)*	72.3 (+7.6 ↑)*	72.6 (+7.8 ↑)*	82.6 (+10.1 ↑)*
HellaSwag ($r = 0.3$)	NON-BIJECTIVE	31.7	29.7	30.7	29.7	30.9	33.1
	BIJECTIVE	34.2 (+2.5 ↑)*	31.7 (+2.0 ↑)	34.1 (+3.4 ↑)*	31.9 (+2.2 ↑)*	31.6 (+0.7 ↑)	33.9 (+0.8 ↑)
WinoGrande ($r = 0.1$)	NON-BIJECTIVE	54.9	53.2	53.7	53.7	53.3	54.3
	BIJECTIVE	56.3 (+1.4 ↑)	53.8 (+0.6 ↑)*	54.2 (+0.5 ↑)*	55.5 (+1.8 ↑)*	54.6 (+1.3 ↑)*	55.5 (+1.2 ↑)*

Table 2: Llama3.1 8B accuracies (reported in %) on different datasets with varying numbers of ICL examples under BIJECTIVE vs. NON-BIJECTIVE ciphers. The numbers inside the parenthesis shows the change from NON-BIJECTIVE to BIJECTIVE cipher. Statistically significant gains are indicated via *.

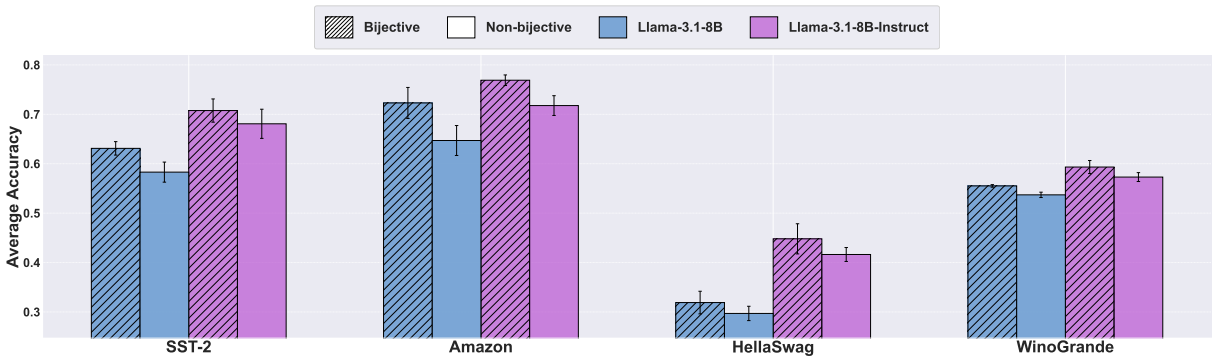

Figure 3: Accuracy comparison of Llama-3.1-8B and Llama-3.1-8B-Instruct on four datasets under BIJECTIVE and NON-BIJECTIVE ciphers with 20-shot (§4.4). **Both aligned and non-aligned models achieve similar relative improvements when solving tasks encoded with a BIJECTIVE cipher, compared to those encoded with NON-BIJECTIVE ciphers**.

4.6 Analysis: Probing Representations

To examine how LLMs process ciphered inputs, we use Logit Lens (nostalgebraist, 2020) to probe their intermediate layer representations. Logit Lens takes token embeddings from intermediate layers and decodes them using the final LM head. We conduct this probing on the Amazon sentiment dataset using Llama 3.1 8B.

Selecting tokens for probing: We first pick 600 most frequent tokens in the demo set after filtering out tokens other than verbs, nouns and adjectives, using NLTK (Bird et al., 2009). We randomly sample 30 tokens from them as the "original tokens". We then randomly sample another 30 tokens from the remaining 570 tokens as the "substituted tokens", each of which has a one-to-one correspondence with the original tokens. **Token substitution:** For BIJECTIVE cipher, we create a bijection between the 30 original tokens and the selected 30 substitution tokens, creating a mapping for the original tokens to be substituted. For NON-BIJECTIVE cipher, we substitute each occurrence of each original token, by a randomly sampled token from the remaining 570 tokens.

Building ciphered inputs: For each original token t' (the token to be ciphered), we sample 15 examples from the demo pool that contain t', and apply our two substitution ciphers to build the ciphered prompt. Given the positions of original tokens $P = (p_1, p_2, ..., p_n)$, we apply the Logit Lens and observe embeddings at positions $P' = (p_1 - 1, p_2 - 1, ..., p_n - 1)$ (i.e., one position prior) to find the ranks of original tokens and "substituted tokens". This gives us an understanding of how the model changes its preference between original and substituted tokens. We quantify this notion as the rank difference (original token rank - substitution token rank):

$$\text{rank-diff} = \text{rank}(t_j) - \text{rank}(c(t_j)), \qquad (1)$$

where rank denotes the position of a given token in the model's softmax score over the vocabulary set.

LLM representations favor substituted tokens in BIJECTIVE cipher: For BIJECTIVE cipher (Fig.5; left) as the model observes more substitutions, the

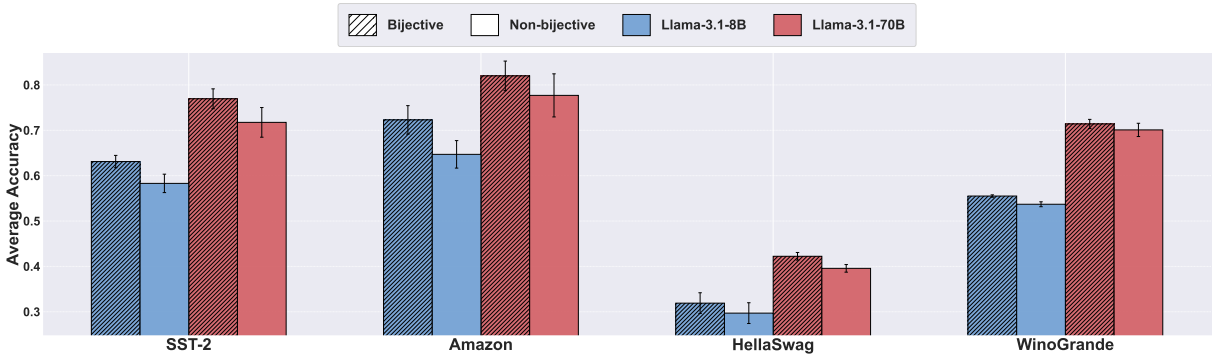

Figure 4: Accuracy comparison of Llama-3.1-8B and Llama-3.1-70B on four datasets under BIJECTIVE and NON-BIJECTIVE ciphers with 20-shot (§4.5).

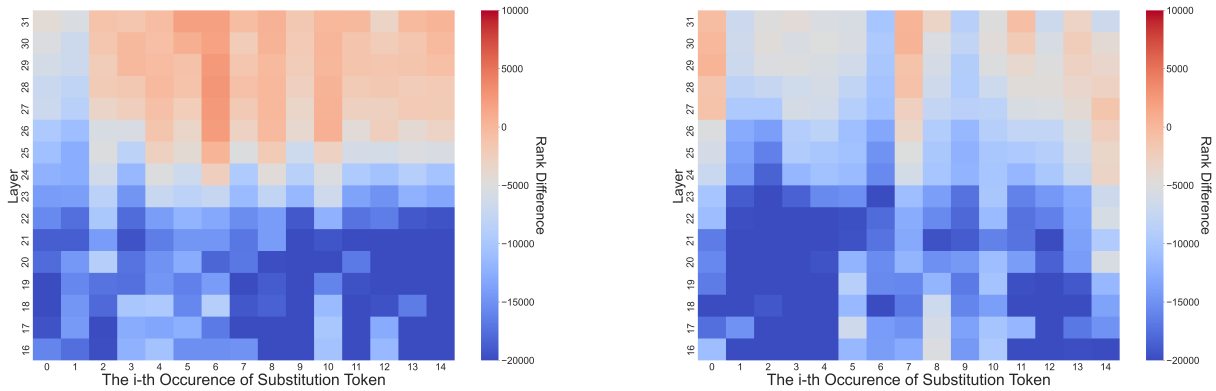

Figure 5: x-axis indicates the i-th occurrence of ciphered tokens in the Llama 3.1 context. y-axis indicates the rank difference (Eq 1). Positive values (red) indicate the model's preference for substituted tokens over original ones. In the BIJECTIVE cipher (left), we see a preference that favors substituted tokens. However, there is no clear preference in the NON-BIJECTIVE cipher (right).

rank difference changes from negative to positive (in deeper layers, where the model interpreting with LogitLens is more meaningful). Consistently, the model gives a higher rank to the substituted tokens than the original tokens, suggesting that the model starts to understand the cipher. In contrast, there is no trend for NON-BIJECTIVE cipher (Fig.5; right) as there is nothing to decipher.

5 Related Work

Dual operating modes of ICL: Min et al. (2022) showed the disconnect between "learning" and the content of in-context demonstrations (lack of task "learning"). This motivated following works to identify two primary modes of operation for In-Context Learning (ICL): *task retrieval* (TR), which involves recalling patterns previously encountered in pre-training data, and *task learning* (TL), which involves learning new patterns on-the-fly that were not seen during pre-training. Some studies emphasize TR by exploring the factual recall capabilities of ICL (Sun et al., 2023; Golchin et al., 2024; Han et al., 2023; Zhao, 2023; Reddy, 2023; Dankers and Titov, 2024), providing insights into how LLMs memorize pre-training data, thus facilitating TR. Other studies (Lin and Lee, 2024; Song et al., 2024; Nafar et al., 2024; Anand et al., 2024) focus on simplified datasets (e.g., linear regression) or architectures (e.g., shallow transformers), which differ from our focus. Other studies focus on the TL capabilities of the model, but use simplified datasets (e.g., linear regression) or architectures (e.g., shallow transformers) (Lin and Lee, 2024; Song et al., 2024; Nafar et al., 2024; Anand et al., 2024). In contrast, our method is aimed at real datasets and real LLMs. Additionally, Pan et al. (2023); Wang et al. (2024) have attempted to separate TR and TL through *output* intervention by replacing labels with abstract symbols like numbers or letters. However, it remains uncertain whether using abstract labels effectively eliminates the influence of TR in ICL. Many human-readable tasks may have inherent priors embedded in the pre-training datasets, suggesting that LLMs might

still use inputs and prompt structures to infer the task, thereby engaging in implicit task retrieval. Our approach proposes an alternative method for quantifying TL by intervening in the *input* space.

Ciphers and their use in AI: Substitution ciphers are studied in NLP for their potential to decipher lost languages without parallel corpora (Knight et al., 2006; Ravi and Knight, 2008, 2011; Dou and Knight, 2012; Berg-Kirkpatrick et al., 2013; Pourdamghani and Knight, 2017; Nuhn et al., 2013; Berg-Kirkpatrick and Klein, 2011; Corlett and Penn, 2010; Aldarrab and May, 2020, *inter alia*). For example, Ravi and Knight (2011) propose a Bayesian approach combining n-gram models and dictionaries for efficient sampling-based decipherment. Deterministic methods also exist, using optimization or heuristics (Peleg and Rosenfeld, 1979; Ganesan and Sherman, 1993; Olson, 2007). Yuan et al. (2023) is the only work we know applying ciphers to LLMs (GPT-4) in the context of safety.

6 Discussion and Conclusion

BIJECTIVE cipher is not a *single* task. The proposed ciphers are a broad reformulation mechanism of *existing* tasks. The underlying task can be *any task* chosen by the user. Our method offers a general-purpose framework for task reformulation that enables us to probe the boundary between memorization and generalization.

Does BIJECTIVE cipher guarantee measuring *only* "learning"? Achieving a perfect distinction between "learning" and "retrieval" may be unattainable, as any learning inherently involves non-zero level of retrieval (e.g., language understanding). Our framework provides a method to quantify learning, by analyzing the difference between how LLMs process a random but learnable bijection, vs non-bijective noise. Though understanding the complementarity of these approaches and success at quantifying pure learning remains to be further understood in future work.

Do the modest gains of BIJECTIVE cipher indicate that the weakness of "learning" in ICL? Not necessarily. The proposed re-encoding of ICL transforms tasks into more complex problems that are inherently more challenging to solve. This is a feature, not a bug, as it allows us to argue that such esoteric encoding tasks reduce the potential confounding effect of retrieval. However, the side effect is that this increased difficulty in task re-encoding results in smaller gains. The key point is that there are consistent positive gains between the BIJECTIVE and NON-BIJECTIVE settings. The magnitude of this gap is a secondary consideration and is likely to change with future innovative methods for re-encoding tasks.

Can your results be due to data contamination? Our work is motivated by the same issue. Data contamination makes it difficult to attribute the success of ICL to "retrieval" (from pre-training) vs "learning" (from in-context demonstrations, without seeing them a priori). A reasonable approach to measure the latter (and mitigate the former) is through randomized tasks. The point of our study is to substitute the given tasks with randomly generated bijection tokens, which makes it impossible for any model to have memorized them. We report the *difference* in performance with bijective vs non-bijective ciphering and de-emphasize any absolute performance numbers which could have resulted from memorization of the original task.

To measure TL, why don't we just evaluate LLMs on "novel" tasks? There is currently no straightforward way to define task "novelty". Prior work has shown that LLM performance correlates strongly with the presence of tasks in the pretraining data (Razeghi et al., 2022; Mallen et al., 2023). To quantify novelty, one would need to either (i) perform large-scale fuzzy matching against pretraining corpora, or (ii) recast tasks into an equivalent representation that is unlikely to have been encountered during pretraining. Few works have tried (i) and have shown some success, but we also know that it's brittle and challenging. Hence, our work focuses on (ii).

Conclusion: We introduced ICL CIPHERS, a class of cryptography text transformations designed to evaluate novel task learning capabilities of LLMs. We show that LLMs exhibit the capacity to decipher these novel tasks during inference. This evidence indicates LLMs' ability to learn novel tasks outside of their pre-training corpus. The exact mechanism of this "learning" remains an active area of study. Understanding this mechanism holds the potential to unleash novel problem-solving capabilities of LLMs.

Limitations

We discuss the potential limitations of our work:

Deviation from natural language: Ciphered text generated diverges from natural language. While this is useful to assess LLMs' TL capabilities, it may also make the task excessively challenging for them. It is possible there might be alternative ways to measure learning in a way that maintains the naturalness of the tasks.

More models and datasets: Although we evaluated 24 settings (six models × four datasets), expanding our study to include more and larger models would strengthen our findings. The largest model we tested was Llama 3.1 70B, due to limited computing resources. Additionally, we did not evaluate large, aligned models such as GPT-4-o1, or Gemini. Anecdotal evidence suggests that aligned models may lose their ability to follow in-context demonstrations (Fu et al., 2022), a crucial aspect of our task definition. However, we acknowledge that our task could potentially be adapted into a task description or instruction format suitable for aligned models, which deviates from our current setting and could be explored in future work. It would also be interesting to evaluate ICL CIPHERS on various pre-training checkpoints to better understand how ICL "learning" emerges through pre-training.

Deeper interpretability analysis: In terms of interpretability analysis, we experimented with several approaches (e.g., PatchScope (Ghandeharioun et al., 2024)) but found success only with the simplest method, the Logit Lens. More advanced interpretability analyses could provide deeper insights into the underlying mechanisms, offering a clearer understanding of the processes involved.

We recognize these as areas for further exploration and encourage future research to address these limitations.

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2022. What learning algorithm is in-context learning? investigations with linear models. In *International Conference on Learning Representations* (ICLR).

Nada Aldarrab and Jonathan May. 2020. Can sequence-to-sequence models crack substitution ciphers? *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 7226–7235.

Suraj Anand, Michael A Lepori, Jack Merullo, and Ellie Pavlick. 2024. Dual process learning: Controlling use of in-context vs. in-weights strategies with weight forgetting. *arXiv preprint arXiv:2406.00053*.

Taylor Berg-Kirkpatrick, Greg Durrett, and Dan Klein. 2013. Unsupervised transcription of historical documents. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 207–217.

Taylor Berg-Kirkpatrick and Dan Klein. 2011. Simple effective decipherment via combinatorial optimization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 313–321.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems* (NeurIPS).

Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. 2022. Data distributional properties drive emergent in-context learning in transformers. *Advances in Neural Information Processing Systems* (NeurIPS), 35:18878–18891.

Eric Corlett and Gerald Penn. 2010. An exact a* method for deciphering letter-substitution ciphers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1040–1047.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada. Association for Computational Linguistics.

Verna Dankers and Ivan Titov. 2024. Generalisation first, memorisation second? memorisation localisation for natural language classification tasks. *arXiv preprint arXiv:2408.04965*.

Qing Dou and Kevin Knight. 2012. Large scale decipherment for out-of-domain machine translation. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 266–275.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Wikimedia Foundation. Wikimedia downloads.

Yao Fu, Hao Peng, and Tushar Khot. 2022. How does gpt obtain its ability? tracing emergent abilities of language models to their sources. *Yao Fu's Notion*.

Ravi Ganesan and Alan T Sherman. 1993. Statistical techniques for language recognition: An introduction and guide for cryptanalysts. *Cryptologia*, 17(4):321–366.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems* (NeurIPS), 35:30583–30598.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A unifying framework for inspecting hidden representations of language models. In *Forty-first International Conference on Machine Learning*.

Shahriar Golchin, Mihai Surdeanu, Steven Bethard, Eduardo Blanco, and Ellen Riloff. 2024. Memorization in in-context learning. *arXiv preprint arXiv:2408.11546*.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, et al. 2024. Olmo: Accelerating the science of language models. *Preprint*.

Michael Hahn and Navin Goyal. 2023. A theory of emergent in-context learning as implicit structure induction. *arXiv preprint arXiv:2303.07971*.

Xiaochuang Han, Daniel Simig, Todor Mihaylov, Yulia Tsvetkov, Asli Celikyilmaz, and Tianlu Wang. 2023. Understanding in-context learning via supportive pretraining data. In *Annual Meeting of the Association for Computational Linguistics* (ACL), pages 12660–12673.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2024. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*.

Juno Kim and Taiji Suzuki. 2024. Transformers learn nonlinear features in context: Nonconvex mean-field dynamics on the attention landscape. *International Conference on Machine Learning* (ICML).

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 499–506.

Shuai Li, Zhao Song, Yu Xia, Tong Yu, and Tianyi Zhou. 2023. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*.

Ziqian Lin and Kangwook Lee. 2024. Dual operating modes of in-context learning. In *International Conference on Machine Learning* (ICML).

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Annual Meeting of the Association for Computational Linguistics* (ACL).

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. In *Annual Meeting of the Association for Computational Linguistics* (ACL).

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work? *arXiv preprint arXiv:2202.12837*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022. Reframing instructional prompts to gptk's language. In *Annual Meeting of the Association for Computational Linguistics* (ACL) - *Findings*.

Aaron Mueller, Albert Webson, Jackson Petty, and Tal Linzen. 2024. In-context learning generalizes, but not always robustly: The case of syntax. In *Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL), pages 4761–4779.

Aliakbar Nafar, Kristen Brent Venable, and Parisa Kordjamshidi. 2024. Learning vs retrieval: The role of in-context examples in regression with llms. *arXiv preprint arXiv:2409.04318*.

nostalgebraist. 2020. Interpreting gpt: The logit lens.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2013. Beam search for solving substitution ciphers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1568–1576.

Edwin Olson. 2007. Robust dictionary attack of short simple substitution ciphers. *Cryptologia*, 31(4):332–342.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Shmuel Peleg and Azriel Rosenfeld. 1979. Breaking substitution ciphers using a relaxation algorithm. *Communications of the ACM*, 22(11):598–605.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Advances in Neural Information Processing Systems* (NeurIPS).

Steven T Piantadosi. 2014. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic bulletin & review*, 21:1112–1130.

Nima Pourdamghani and Kevin Knight. 2017. Deciphering related languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2513–2518.

Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 812–819.

Sujith Ravi and Kevin Knight. 2011. Bayesian inference for zodiac and other homophonic ciphers. In *Annual Meeting of the Association for Computational Linguistics* (ACL), pages 239–247.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP) - *Findings*.

Gautam Reddy. 2023. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *International Conference on Learning Representations* (ICLR).

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WINOGRANDE: an adversarial winograd schema challenge at scale. In *Conference on Artificial Intelligence* (AAAI).

Lingfeng Shen, Aayush Mishra, and Daniel Khashabi. 2024. Do pretrained transformers learn in-context by gradient descent? In *International Conference on Machine Learning* (ICML).

Seongjin Shin, Sang Woo Lee, Hwijeen Ahn, Sungdong Kim, Hyoung Seok Kim, Boseop Kim, Kyunghyun Cho, Gichang Lee, Woomyoung Park, Jung Woo Ha, et al. 2022. On the effect of pretraining corpora on in-context learning by a large-scale language model. In *Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL).

Suzanna Sia, David Mueller, and Kevin Duh. 2024. Where does in-context translation happen in large language models. *arXiv preprint arXiv:2403.04510*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), pages 1631–1642.

Jiajun Song, Zhuoyan Xu, and Yiqiao Zhong. 2024. Out-of-distribution generalization via composition: a lens through induction heads in transformers. *arXiv preprint arXiv:2408.09503*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research* (TMLR).

Zhiqing Sun, Xuezhi Wang, Yi Tay, Yiming Yang, and Denny Zhou. 2023. Recitation-augmented language models. *International Conference on Learning Representations* (ICLR).

Gemma Team. 2024a. Gemma.

Qwen Team. 2024b. Qwen2.5: A party of foundation models.

Robert Vacareanu, Vlad-Andrei Negru, Vasile Suciu, and Mihai Surdeanu. 2024. From words to numbers: Your large language model is secretly a capable regressor when given in-context examples. In *Conference on Language Modeling* (COLM).

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *International Conference on Learning Representations* (ICLR), pages 35151–35174.

Xiaolei Wang, Xinyu Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Investigating the pre-training dynamics of in-context learning: Task recognition vs. task learning. *arXiv preprint arXiv:2406.14022*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-Instruct: Aligning Language Model with Self Generated Instructions. In *Annual Meeting of the Association for Computational Linguistics* (ACL).

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. 2023. Gpt-4 is too smart to be safe: Stealthy chat with llms via cipher. *arXiv preprint arXiv:2308.06463*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. 2023. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*.

Jiachen Zhao. 2023. In-context exemplars as clues to retrieving from large associative memory. *arXiv preprint arXiv:2311.03498*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning* (ICML), pages 12697–12706.

Supplemental Material

A Additional Related Work

Alternative explanations of ICL: Since the discovery of ICL (Brown et al., 2020), numerous studies have explored it across various contexts (Zhao et al., 2021; Min et al., 2022; Mishra et al., 2022; Han et al., 2023; Wang et al., 2023; Sia et al., 2024; Vacareanu et al., 2024; Mueller et al., 2024). For example, Perez et al. (2021); Lu et al. (2022); Mishra et al. (2022) demonstrated ICL's sensitivity to the selection and sequence of demonstrations, while Shin et al. (2022); Razeghi et al. (2022) highlighted its sensitivity to the frequency and size of the relevant pre-training corpus. Another research direction seeks to elucidate the mechanisms behind ICL. Xie et al. (2021) described ICL as implicit Bayesian inference, where ICL demonstrations are mapped to a latent concept (task) learned during pre-training. Other works have attempted to explain ICL as a form of implicit optimization (gradient descent and its variants) (Garg et al., 2022; Zhang et al., 2023; Dai et al., 2023; Von Oswald et al., 2023; Li et al., 2023), though the applicability of these formalisms to real LLMs is debated (Shen et al., 2024). A few studies aim to understand how ICL emerges in LLMs. Hahn and Goyal (2023) suggested that the compositional structure of natural language leads to emergent in-context learning, while other works (Chan et al., 2022) propose that certain distributional properties in the pre-training data may give rise to ICL. Although these studies offer varying perspectives into the origin and functioning nature of ICL, we propose to disentangle TL and TR components of ICL by observing LLMs' behavior on randomly generated bijections vs. non-bijection noise.

B Additional Experimental Details

B.1 Preserved Tokens

For Llama 3.1, we preserve the tokens whose ids range from 0 to 255, 128000 to 128256. For Qwen 2.5, we preserve the tokens whose ids range from 0 to 255, 151643 to 151664. For OLMo, we preserve the tokens whose ids range from 0 to 244, 50254 to 50279. For Gemma 2, we preserve the tokens whose ids range from 0 to 472, 255968 to 255999. For all the models, we preserve the spaces and underlines to ensure the framework of each task. For example, in the WinoGrande dataset, LLMs are asked to predict the pronouns in a sentence, where the original pronouns are replaced by an underline.

B.2 Handling of White Space

LLMs encode the spaces between words differently depending on their tokenization. Gemma 2 uses a special underline to represent a space, while Llama 3.1 , QWen 2.5 and OLMo uses 'Ġ'. There are usually two versions of the same word – with or without a space before it, which corresponds to two different tokens. Take Llama 3.1 for example, the encoded id of "is" is 285 while that of "Ġis" is 374. We name tokens containing a space at the beginning as "space tokens" and the others as "non-space tokens". To avoid disturbing spaces in the original text, which may confuse the model, we constrain the shuffling to be within their space/non-space sets.

B.3 Design choices for ICL CIPHERS

In Tab.3, we explain our design strategies for choosing priority sampling (in selecting demonstrations from the demo pool) and zipfian shuffling (in choosing the mapping c).

Strategies for ...	Variant 1	Variant 2
selecting (sampling) demonstrations	**Priority**: select demonstrations that contain the target substitution in the test example ✓	**Non-priority**: select demonstrations randomly ✗
choosing the token mapping c	**Zipfian**: c maps tokens of similar frequency (popularity) among each other ✓	**Non-Zipfian**: c maps tokens irrespective of their frequency (popularity) ✗

Table 3: Design choices for experiments in ICL CIPHERS discussed in §3.1.

B.4 Datasets

For SST-2, HellaSwag and WinoGrande no label provided for the test set. Therefore, we use their validation set instead.

SST-2: We use its validation set as our test set, which has size of 872. Its training set, which contains 67.3k examples, is used as the demo pool.

Amazon: To fit the Amazon dataset into binary sentiment classification framework, we filter ratings 4-5 as positive and 1-2 as negative (discard rating 3). We focus on reviews under the the "All_Beauty" category in our experiments. We sample 144k positive and negative samples to build the demo pool; and 500 other positive and negative examples as the test set.

HellaSwag: We use its validation set as our test set, which contains 444 positive examples and 428 negative examples (872 examples in total). Its training set, which contains 38K positive examples and 30k negative examples, is used as the demo pool.

We randomly sample 1k examples from the validation set as our test set. We use its training set as the demo pool, which contains 40k examples.

WinoGrande: We use its dev set as the test set, which contains 1267 examples. Its xl training set is used as demo pool, which has 40k examples.

B.5 Prompt Template

We don't include any instructions in our prompt. For SST-2 and Amazon, we use the following prompt template:

Input: {*input_demo*}

Output: {*label_demo*}

...

Input: {*input_test*}

where {*input_demo*} and {*label_demo*} are the input text and sentiment labels of demonstrations, and {*input_test*} is the input text of test case.

For HellaSwag and WinoGrande, we use the following prompt template:

Question: {*question_demo*}

Options: {*options_demo*}

Answer: {*answer_demo*}

...

Question: {*question_test*}

Options: {*options_test*}

where *question_demo*}, *options_demo*} and {*answer_demo*} are the questions, options and correct answers of demos, and *question_test*} and *options_test*} are the question and option of the test case.

C Example Inputs/Outputs

Here we display the example inputs/outputs on all the four datasets. Note that in our experiments the original inputs are not included in the prompts.

Dataset: SST-2; Model: QWen 2.5 ; Cipher: BIJECTIVE; Shuffle Rate: 0.6

Ciphered Input: been sc Mil Swift the Inch for pen Venezuela Moody
Original Input: been sent back to the tailor for some major alterations
Output: negative

Ciphered Input: is born Slovenia of an Platform San sitcom involved also Sr implementedecture embarrassed Swift Malay you reach for the tissues Confederate
Original Input: is born out of an engaging storyline , which also is n't embarrassed to make you reach for the tissues .
Output: positive
...

Ciphered Test Input: allows us Swift hope Esc implementedolan Sr poised Swift cheating a Venezuela career Mr a assembled Kann steak filmmaker Confederate
Original Test Input: allows us to hope that nolan is poised to embark a major career as a commercial yet inventive filmmaker .

Dataset: Amazon ; Model: Gemma 2 ; Cipher: BIJECTIVE; Shuffle Rate: 0.6

Ciphered Input: didnSUwell really notice anything mob. I sink it householder substance Woodward Bean Simple Woodward Senior Caldwell Snowyyn Ato was instance.
Original Input: didn't really notice anything special. I bought it because of the reviews and the price but honestly, I was disappointed.
Output:negative

Ciphered Input: Item arrived regions principle unrest neighbours']modern /><modern urchatosyn Woodward item was calcium steamer principle Counter cap rendering Woodward cover ent since it periodsSUwell Fam Arch anymore Simple iconicBer bottom Simple consequently']modern /><modern urchofficial was wrapped dentist regions principle padded envelope.
Original Input: Item arrived in a quick manner.

However, the item was received with a damaged cap rendering the cover useless since it won't snap on anymore and dented bottom and top.

It was wrapped tightly in a padded envelope.
Output:negative
...

Ciphered Test Input: tried it for cosmetic qualifications perimeter a day spaŏ0f2 didnPervers Tehran workil
Original Test Input: tried it for cosmetic procedures in a day spa; didn't really work.

Dataset: Hellaswag; Model: OLMo ; Cipher: BIJECTIVE; Shuffle Rate: 0.3

Ciphered Question: Ter Back sits million titled with his Board effective on the keys. the Back
Original Question: A man sits a piano with his hands placed on the keys. the man
Ciphered Options: (1) begins playing the titled.\n(2) Carlos the keys with million malorn.\n(3) beats the titled in million benefitedmic thought.\n(4) increases the play for playing.\n
Original Options: (1) begins playing the piano.\n(2) hits the keys with a mallet.\n(3) beats the piano in a rhythmic beat.\n(4) increases the volume for playing.\n
Answer: (1)
...
Ciphered Question: People are noted on the street. million Back
Original Question: People are running on the street. a man
Ciphered Options: (1) is wearing poetilts.\n(2) limited million drink out Wars million After presidents.\n(3) negotiating into million encourages Wars fire.\n(4) limited million high jump in million Chris competition.\n
Original Options: (1) is wearing stilts.\n(2) takes a drink out of a water bottle.\n(3) jumps into a pile of fire.\n(4) takes a high jump in a bar competition.\n

Dataset: WinoGrande ; Model: Llama 3.1 ; Cipher: BIJECTIVE; Shuffle Rate: 0.3

```
Ciphered Question: Estonia ferry that my parents story tied I permanent in Johnston permanent
Stadium partners bla than my house now because the _ permanent anchored.
Original Question: The home that my parents had when I was in school was a lot nicer than my
house now because the _ was sophisticated.
Ciphered Options: (1) ferry, (2) house
Original Options: (1) home, (2) house
Answer:(1)
...
Ciphered Question: Sarah permanent Stadium much better Chart than Maria so _ always got the
easier cases.
Original Question: Sarah was a much better surgeon than Maria so _ always got the easier cases.
Ciphered Options: (1) Sarah, (2) Maria
Original Options: (1) Sarah, (2) Maria
```

D Additional Related Work

Empirical understanding of ICL: Ever since In-Context Learning was discovered (Brown et al., 2020), multiple works have studied it under diverse settings (Zhao et al., 2021; Min et al., 2022; Mishra et al., 2022; Han et al., 2023; Wang et al., 2023; Sia et al., 2024; Vacareanu et al., 2024; Mueller et al., 2024). For instance, Srivastava et al. (2023) benchmarked ICL under multiple tasks and models; Perez et al. (2021); Lu et al. (2022) showed the sensitivity of ICL to the choice of demonstrations and their orderings; Shin et al. (2022); Razeghi et al. (2022) showed the sensitivity of ICL performance to the frequency and size of the relevant pre-training corpus. These works have made useful observations that allow us to better use this elusive quality of LLMs.

Functional nature of ICL: A more recent line of study aims to understand how ICL actually works in LLMs. Multiple works have compared ICL with implicit optimization (specifically gradient descent) (Garg et al., 2022; Zhang et al., 2023; Dai et al., 2023; Akyürek et al., 2022; Von Oswald et al., 2023; Li et al., 2023; Kim and Suzuki, 2024). This line of work claims that Transformers can meta-learn to perform optimization of internal models given a set of demonstrations. However, their study setup with toy transformers does not align with how LLMs are trained as shown by Shen et al. (2024). Moreover, this line of study does not explain the TR capabilities of LLMs.

Forces that lead to ICL: Few works try to understand *how ICL emerges in LLMs*. Xie et al. (2021) explained ICL as implicit Bayesian inference, which maps a ICL demonstrations to a latent concept (task) learned via pre-training. Hahn and Goyal (2023) posited that compositional structure in natural language gives rise to emergent in-context learning. Other works (Chan et al., 2022) theorize more distributional properties in the pre-training data, that may give rise to ICL. Many of these works explain some properties of ICL but fail at others. The exact origin of ICL in LLMs still remains an active area of study.

E Priority vs. Non-Priority Sampling

Fig.2 shows peformance of LLaMa 3.1 8B on Amazon dataset with priority sampling. Fig.7 and Fig.8 shows peformance of LLaMa 3.1 8B on SST-2 and Amazon datasets with non-priority sampling. Comparing with Fig.6 and Fig.2, they demonstrate similar trends but the performances are more unstable due to the randomness of non-priority sampling. Therefore, we use priority sampling throughout our experiments for more steady results.

F Pretrained-only vs. Aligned Models

Table 4 shows the performances of Llama3.1-8B-Instruct on different datasets. Comparing with its pretrained-only version (Table 2), it demonstrates better performances. However, their gaps between BIJECTIVE and NON-BIJECTIVE ciphers are on-par.

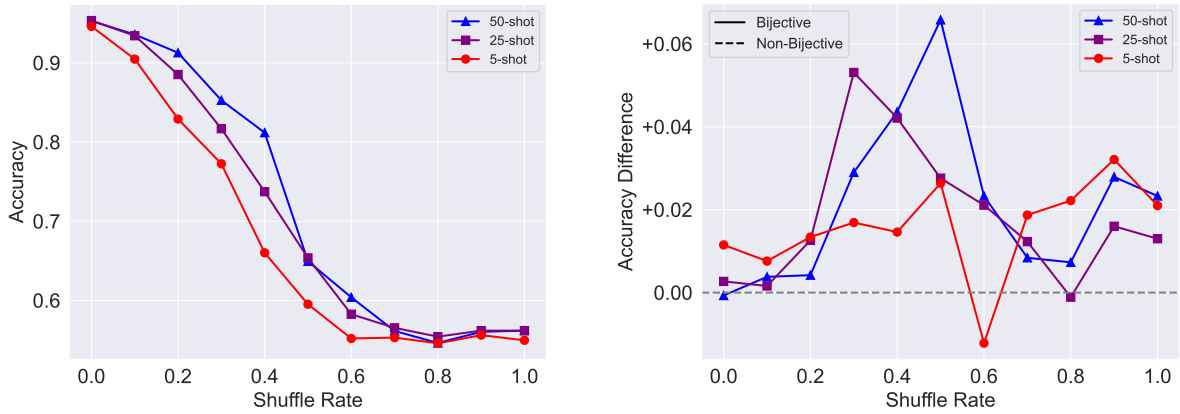

Figure 6: Llama 3.1 8B performance on SST-2 dataset, which shows similar trends with Fig.2. **Left:** Under the BIJECTIVE cipher, accuracy decreases smoothly as the shuffle rate increases, highlighting the difficulty in interpreting the ciphered text. Accuracy also increases with more demonstrations, suggesting the model's ability to solve BIJECTIVE cipher. **Right:** y-axis shows the accuracy gap between BIJECTIVE and NON-BIJECTIVE ciphers. For very high shuffle rates (e.g, > 0.7), the task become very hard to understand and solve (for the model and even humans) as it becomes ill-defined.

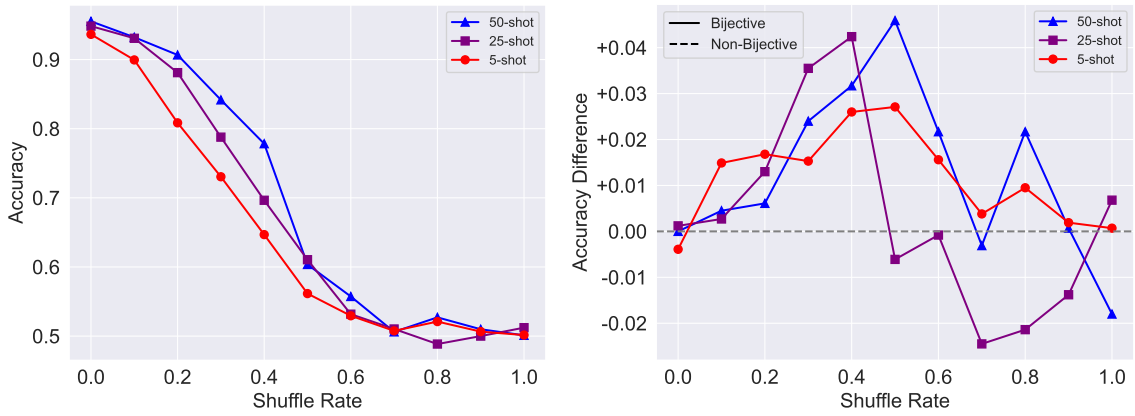

Figure 7: Peformance of Llama 3.1 8B on SST-2 dataset with non-priority sampling, comparing with Fig.6. **Left:** The accuracies under BIJECTIVE cipher. **Right:** The y-axis displays the accuracy gap between BIJECTIVE and NON-BIJECTIVE ciphers.

G Small vs. Large Model

Table 5 shows the performances of Llama3.1-70B on different datasets. Comparing with Llama3.1-8B (Table 2), it demonstrates better performances. However, their differences in gaps between BIJECTIVE and NON-BIJECTIVE ciphers are mixed.

H Significance of Results

To determine if the gaps between BIJECTIVE and NON-BIJECTIVE ciphers are significant, we conduct McNemar's test (McNemar, 1947). Table 6, Table 7, Table 8 and Table 9 show the computed p-values for Table 1, Table 2, Table 4 and Table 5 respectively. The gap is regard as significant if its corresponding p-value is no larger than 0.05.

I Restricting the Space of Cipher

We notice that the gaps on HellaSwag and WinoGrande are smaller than those in SST-2 and Amazon. The reason behind it could be the complexity of these two datasets, which could impact the model's ability to solve the ciphers. To verify this, we constrain the vocabulary shuffling to only nouns on these two datasets.

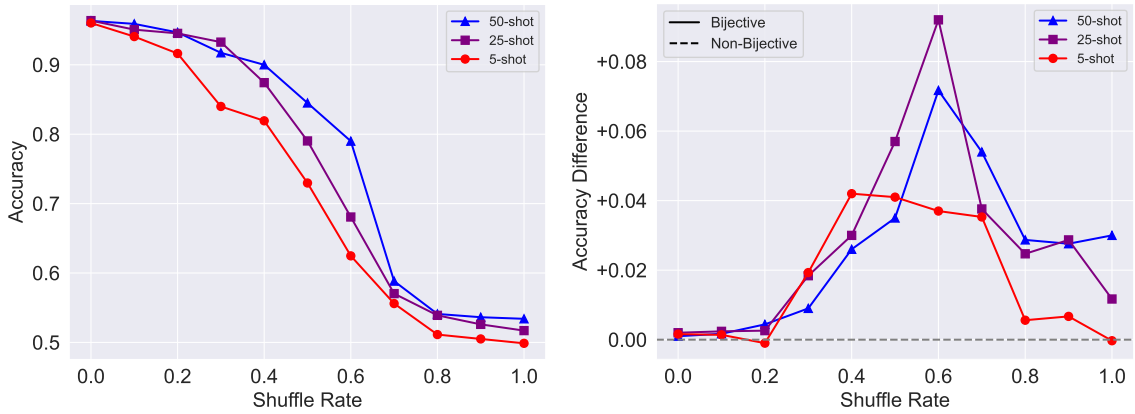

Figure 8: Peformance of Llama 3.1 8B on Amazon dataset with non-priority sampling, comparing with Fig.2. **Left:** The accuracies under BIJECTIVE cipher. **Right:** The y-axis displays the accuracy gap between BIJECTIVE and NON-BIJECTIVE ciphers.

Shots → Dataset (shuffle rate)↓	Cipher	Model: Llama 3.1 8B Instruct					
		5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.5$)	NON-BIJECTIVE	65.5	66.5	68.9	68.1	67.5	65.5
	BIJECTIVE	69.8 (+4.3 ↑)*	70.8 (+4.3 ↑)*	72.4 (+3.5 ↑)*	70.8 (+2.7 ↑)*	70.0 (+2.5 ↑)*	71.8 (+6.3 ↑)*
Amazon ($r = 0.6$)	NON-BIJECTIVE	70.5	80.0	77.3	79.3	80.6	80.5
	BIJECTIVE	75.8 (+5.3 ↑)*	82.7 (+2.7 ↑)*	82.4 (+5.1 ↑)*	82.4 (+3.1 ↑)*	84.6 (+4.0 ↑)*	86.1 (+5.6 ↑)*
HellaSwag ($r = 0.3$)	NON-BIJECTIVE	43.2	43.2	42.3	41.6	41.4	41.0
	BIJECTIVE	44.8 (+1.6 ↑)*	47.5 (+4.3 ↑)*	44.4 (+2.1 ↑)*	44.8 (+3.2 ↑)	45.1 (+3.7 ↑)*	42.4 (+1.4 ↑)*
WinoGrande ($r = 0.1$)	NON-BIJECTIVE	57.4	58.1	55.7	57.3	56.4	57.1
	BIJECTIVE	59.0 (+1.6 ↑)	58.7 (+0.6 ↑)*	57.4 (+1.7 ↑)*	59.3 (+2.0 ↑)*	58.2 (+1.8 ↑)*	57.4 (+0.3 ↑)*

Table 4: Llama3.1 8B Instruct accuracies (reported in %) on different datasets with varying numbers of ICL examples under BIJECTIVE vs. NON-BIJECTIVE ciphers, as comparing to Table 2. The numbers inside the parenthesis shows the change from NON-BIJECTIVE to BIJECTIVE cipher. Statistically significant gains are indicated via *.

Table 10 shows that the changes of gaps between BIJECTIVE and NON-BIJECTIVE ciphers are mixed.

J Further Results on Probing Analysis

To get a clearer vision, we extract the rank difference from the last layer on SST-2, dividing them equally into 5 chunks, as shown in Fig.10. For random substitution, there is not much change for rank difference. For BIJECTIVE substitution, rank difference increase as the chunk number gets bigger. This suggests that as LLM sees more occurence of the substitution token, it learns to use substitution token as the original token, namely solving ICL CIPHERS.

Shots →	Cipher	Model: Llama 3.1 70B					
Dataset (shuffle rate)↓		5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.5$)	NON-BIJECTIVE	64.8	70.3	66.4	71.8	69.8	74.5
	BIJECTIVE	68.9 (+4.1 ↑)*	71.0 (+0.7 ↑)	69.6 (+3.2 ↑)*	76.9 (+5.1 ↑)*	74.4 (+4.6 ↑)*	80.3 (+5.8 ↑)*
Amazon ($r = 0.6$)	NON-BIJECTIVE	73.1	73.7	80.6	77.7	79.3	82.0
	BIJECTIVE	76.5 (+3.4 ↑)*	78.6 (+4.9 ↑)*	84.4 (+3.8 ↑)*	82.0 (+4.3 ↑)*	84.0 (+4.7 ↑)*	85.7 (+3.7 ↑)*
HellaSwag ($r = 0.3$)	NON-BIJECTIVE	42.2	39.1	40.4	39.6	40.6	38.5
	BIJECTIVE	44.2 (+2.0 ↑)*	43.6 (+4.5 ↑)*	43.1 (+2.7 ↑)*	42.2 (+2.6 ↑)*	40.9 (+0.3 ↑)*	41.6 (+3.1 ↑)*
WinoGrande ($r = 0.1$)	NON-BIJECTIVE	65.1	69.5	69.9	70.1	71.0	67.4
	BIJECTIVE	68.6 (+3.5 ↑)*	70.1 (+0.6 ↑)	71.2 (+1.3 ↑)*	71.4 (+1.3 ↑)*	72.2 (+1.2 ↑)*	70.8 (+3.4 ↑)*

Table 5: Llama3.1 70B accuracies (reported in %) on different datasets with varying numbers of ICL examples under BIJECTIVE vs. NON-BIJECTIVE ciphers. The numbers inside the parenthesis shows the change from NON-BIJECTIVE to BIJECTIVE cipher. Statistically significant gains are indicated via *.

Model →	20-shot			
Dataset (shuffle rate) ↓	Llama3.1	Qwen2.5	Olmo	Gemma2
SST-2 ($r = 0.5$)	**0.000**	**0.000**	**0.000**	**0.001**
Amazon ($r = 0.6$)	**0.000**	**0.000**	**0.000**	**0.000**
HellaSwag ($r = 0.3$)	**0.000**	**0.000**	**0.000**	0.663
WinoGrande ($r = 0.1$)	**0.000**	0.084	0.786	0.943

Table 6: Significance results (p-values) of McNemar's test for Table 1. The gap between BIJECTIVE and NON-BIJECTIVE can be regared as significant if its corresponding p-value is no larger than 0.05, which is bolded.

Shots →	Model: Llama 3.1 8B					
Dataset (shuffle rate)↓	5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.5$)	**0.018**	0.205	0.084	**0.000**	**0.020**	**0.000**
Amazon ($r = 0.6$)	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
HellaSwag ($r = 0.3$)	**0.015**	0.627	**0.000**	**0.000**	0.278	0.357
WinoGrande ($r = 0.1$)	0.110	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**

Table 7: Significance results (p-values) of McNemar's test for Table 2. The gap between BIJECTIVE and NON-BIJECTIVE can be regared as significant if its corresponding p-value is no larger than 0.05, which is bolded.

Shots →	Model: Llama 3.1 8B Instruct					
Dataset (shuffle rate)↓	5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.5$)	**0.000**	**0.000**	**0.001**	**0.016**	**0.025**	**0.000**
Amazon ($r = 0.6$)	**0.003**	**0.002**	**0.000**	**0.000**	**0.000**	**0.000**
HellaSwag ($r = 0.3$)	**0.000**	**0.000**	**0.031**	0.081	**0.000**	**0.023**
WinoGrande ($r = 0.1$)	0.067	**0.000**	**0.000**	**0.013**	**0.015**	**0.000**

Table 8: Significance results (p-values) of McNemar's test for Table 4. The gap between BIJECTIVE and NON-BIJECTIVE can be regared as significant if its corresponding p-value is no larger than 0.05, which is bolded.

Shots →	Model: Llama 3.1 70B					
Dataset (shuffle rate)↓	5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.5$)	**0.000**	0.497	**0.006**	**0.000**	**0.000**	**0.000**
Amazon ($r = 0.6$)	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
HellaSwag ($r = 0.3$)	**0.000**	**0.006**	**0.000**	**0.000**	**0.000**	**0.000**
WinoGrande ($r = 0.1$)	**0.000**	0.446	**0.000**	**0.000**	**0.000**	**0.000**

Table 9: Significance results (p-values) of McNemar's test for Table 5. The gap between BIJECTIVE and NON-BIJECTIVE can be regared as significant if its corresponding p-value is no larger than 0.05, which is bolded.

Dataset (shuffle rate)	Cipher	Llama3.1 8B 20-shot	
		All	Noun
HellaSwag (r = 0.3)	NON-BIJECTIVE	29.7	32.1
	BIJECTIVE	31.9 (+2.2 ↑)	33.6 (+1.5 ↑)
WinoGrande (r = 0.1)	NON-BIJECTIVE	53.7	54.3
	BIJECTIVE	55.5 (+1.8 ↑)	56.7 (+2.4 ↑)

Table 10: Llama3.1 8B accuracies (reported in %) with 20-shot demonstrations, under BIJECTIVE and NON-BIJECTIVE ciphers. "All" operates shuffling on all the tokens while "Noun" constrains shuffling to only nouns.

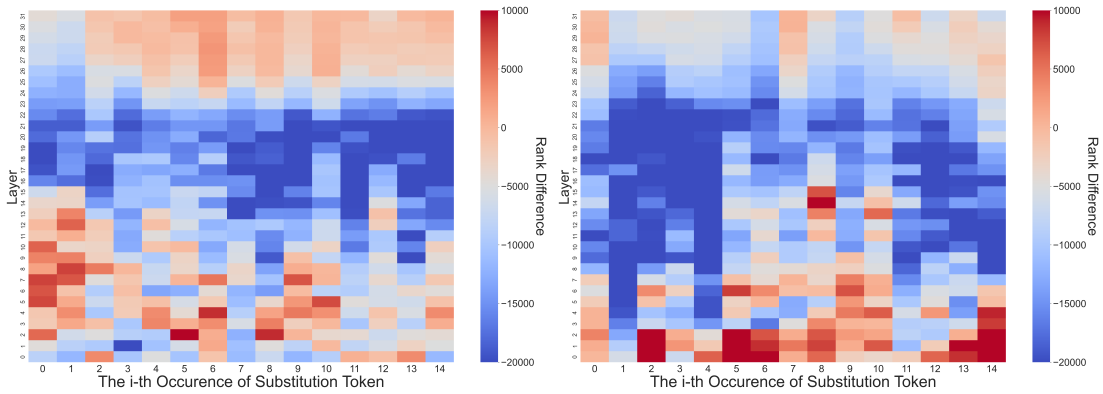

Figure 9: Complete heatmap of original token rank minus substitution token rank on Amazon for Fig.5. **Left:** BIJECTIVE cipher **Right:** NON-BIJECTIVE cipher

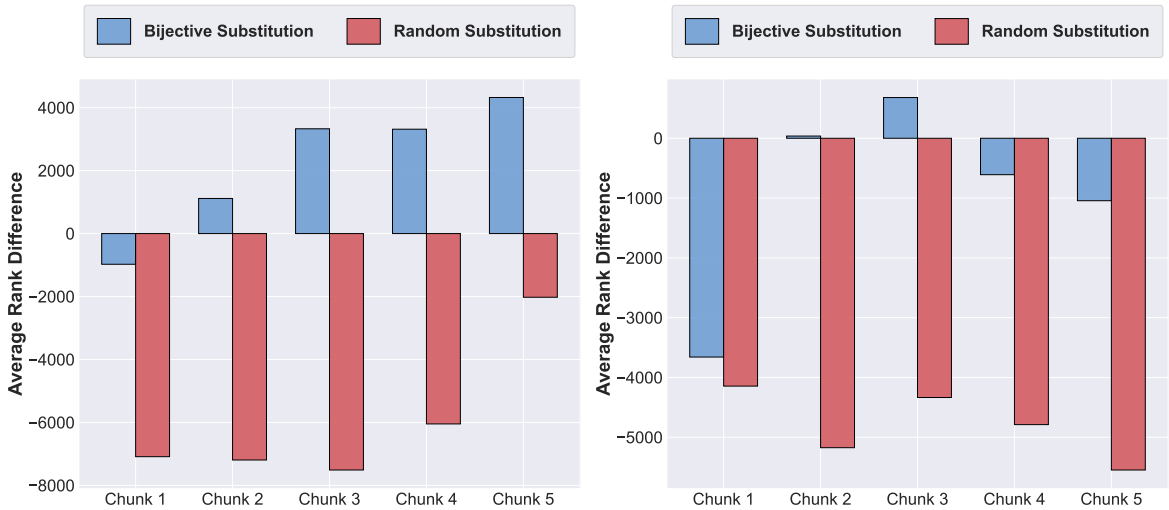

Figure 10: Average rank differences (original token rank - substitution token rank) in SST-2 (left) and Amazon (right) datasets for BIJECTIVE (blue) and NON-BIJECTIVE (red) cipher over 15 occurrences, divided into 5 chunks of size 3. Rank difference serves as a proxy for the model's deciphering ability. Under BIJECTIVE cipher, this ability improves with more exposure to substituted tokens, while NON-BIJECTIVE cipher shows no clear pattern.