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

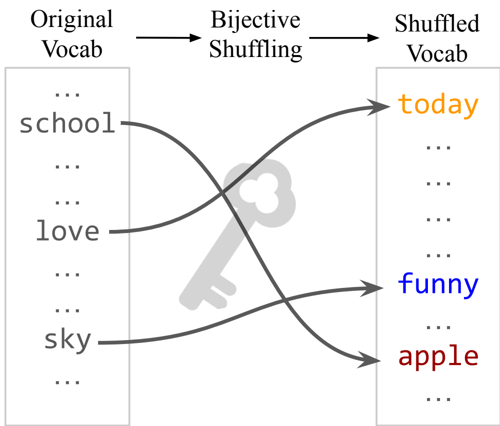
Anonymous ACL submission

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 are capable of solving 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 four models families. Finally, we examine LLMs’ internal representations and identify evidence in their ability to decode the ciphered inputs.



Ciphered In-Context Learning

```
Input: I today my apple! There is …
Output: positive

Input: The funny doesn’t looks so nice
Output: negative

   ⋮

Input: I don’t today looking at the funny
```

Figure 1: An example of ICL CIPHERS, a cryptographic task reformulations framework 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.

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 learning ability has spurred numerous studies to better understand its dynamics. Despite recent efforts (Xie et al., 2021; Min et al., 2022; Srivastava et al., 2023; Shin et al., 2022; Razeghi et al., 2022; Shen et al., 2024), the literature’s understanding of the functional aspects of ICL remains elusive and contentious.

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. Specifically, 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 pretraining. However, it remains unclear whether this label mapping is sufficient to ensure pure TL (i.e., no partial influence of TR). It is conceivable that LLMs could still use the human-readable inputs and prompt structure to deduce the task, thereby performing implicit task retrieval. Fundamentally differentiating TR and TL is hard because it is hard to know whether the learning signal comes from the in-context examples or pretraining data even after modifying input-label mapping suggested in prior work which leaves the inputs intact and leaves the door open for LLM to exploit 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 classic cryptography, applied to task *inputs*. For example, in a sentiment classification task where sentences are assigned to target classes, 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. This random BIJECTIVE shuffling is done before the experiments and remains constant throughout. 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"), as shown in Figure 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, although not easily understood by human eyes, still represents a valid task. 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 results, our working hypothesis is that any gains above the random (NON-BIJECTIVE shuffles should be indicative of TL (as opposed to TR) within ICL.

We evaluate ICL ciphers using 4× pre-trained models across 4× well-known benchmarks and different range few-shot numbers demonstrations. Our empirical results demonstrate that ICL achieves better-than-random performance on ciphered tasks (§5). For example, on the BIJECTIVE ciphered HellaSwag, Llama3.1 (8B) averages 3.6% 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 (§6.4) which reveals promising, albeit weak, trends in their ability to decode the ciphered inputs.

Unlike previous work by (Pan et al., 2023; Wang et al., 2024) that intervenes in task outputs through label shuffling, our approach modifies task inputs. This creates instances less likely to have been encountered in pre-training data, offering an alternative TL indicator by necessitating the LLM to decode ciphers as part of task solving. These perspectives can be seen as complementary, each assessing different aspects of "learning" in ICL.

In conclusion, we propose an alternative method for quantifying "learning" in ICL through the use of substitution ciphers. We establish a framework for evaluating the performance of ICL within this experimental context. Our findings demonstrate evidence of task learning in ICL, both in terms of downstream performance and through interpretability analysis. As far as we know, this is the first work to propose such cryptographic approaches for quantifying genuine "learning" in in-context demonstrations. We hope these insights inspire future research aimed at gaining a deeper understanding of the emergence of ICL in LLMs.

2 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 follow works to identify two primary modes of operation for In-Context Learning (ICL): *task retrieval* (TR), which involves recalling patterns previously encountered in pretraining 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. 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: The problem of deciphering substitution ciphers is studied in NLP as it may provide automatic ways to decipher lost languages without any parallel corpus (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 instance, Ravi and Knight (2011) introduces a Bayesian approach for deciphering substitution ciphers, combining information from letter n-gram language models and word dictionaries to perform efficient sampling-based inference for decipherment results. We also note various optimization-based and heuristic-based computational frameworks that are deterministic in nature for deciphering substitution ciphers (Peleg and Rosenfeld, 1979; Ganesan and Sherman, 1993; Olson, 2007).

We also note the work of Yuan et al. (2023) which is the only work (that we know of) applying ciphers on LLMs (GPT-4) in the context of safety problems, which is a different focus than ours.

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 many of these studies explain certain aspects of ICL, they fall short in others. The precise origins of ICL in LLMs remain an active area of research.

3 Defining ICL CIPHERS

3.1 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 $D_{\text{demo}} = (x_1, y_1, x_2, y_2, \dots, x_n, y_n)$. 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}.

3.2 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 $c : V \to V$ 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 a 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 shffled 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.

3.3 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 Figure 1. Let the input x_i be constituted of K_i tokens, i.e., x_i is the ordered sequence of tokens $(t_1, \dots, t_{K_i})$. For all $t_j = \text{school} \in x_i$ or $x_{\text{test}}, c(t_j) = \text{apple}$. This results in corresponding ciphered inputs x'_i or x'_{test}. Moreover, as c is a bijection, $\exists\, c^{-1}$ such that for all $t_j = \text{apple} \in x'_i$ or $x'_{\text{test}}, c^{-1}(t_j) = \text{school}$. 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 are able to 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 can not 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 has to rely heavily on TL to learn and perform this composite function.

3.4 NON-BIJECTIVE Ciphers

For comparison with BIJECTIVE ciphers (§3.3), 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 (§3.3) 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.

3.5 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 use the gaps between BIJECTIVE (§3.2) and NON-BIJECTIVE (§3.4) ciphers to quantify the "learning" in ICL. 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.

4 Experimental Setup

We evaluate ICL CIPHERS on a range of LLMs and datasets. We then use the difference between the two types of ciphers to quantify a proxy for TL capabilities of these LLMs on various tasks (§3.5).

4.1 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 chunk. Since the pre-training corpora are usually not accessible for LLMs, we use a representative external corpus to approximate the real token frequency. Specifically, we use the Wikipedia (Foundation) to calculate token frequency instead, which is an approximation to the actual token frequency.

Priority sampling of ICL demos: To create an ICL demo set, one way to do it is 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 tokens to be shuffled 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 contain 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 are 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. For each of these cases, we randomly sample a demonstration.

We always use priority sampling (unless otherwise specified). However, in §D we compare priority sampling with non-priority (random) sampling.

Shuffle Rate: The shuffle rate r determines the proportion of tokens that are replaced. 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 nearly 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 §6.1.

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 are in Appendix A.1). Similarly, we avoid disturbing spaces in the original text (details in Appendix A.2).

4.2 Models

We 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`). We don't do experiments on aligned (instruction-tuned or RLHF-ed) models as prior work shows that alignment trades typically hurts in-context learning performance (Fu et al., 2022).

4.3 Datasets

We conduct experiments on four datasets. SST-2 (Socher et al., 2013) and Amazon (Hou et al., 2024, Amazon Reviews 2023) are for binary sentiment classification task. HellaSwag (Zellers et al., 2019) is for sentence completion task, formatted as four-choices QAs. WinoGrande (Sakaguchi et al., 2020) is for pronoun resolution task, formatted as binary-choice QA. For each dataset, we curate a demo pool for sampling ICL demos and a test set contain 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 Appendix.

5 Evidence of Task-Learning in ICL

Table 1 shows the performance of fours LLMs on four datasets ciphered using ICL CIPHERS.

LLMs can decipher BIJECTIVE Ciphers: We see a consistent improvement in the performance of LLMs on BIJECTIVE ciphered inputs over NON-BIJECTIVE ciphered inputs (except for OLMo on SST-2). With fixed shuffle rate and number of demonstrations, any influence of **task retrieval** on the model performance remains the same for both ciphered inputs. However, the consistent gap clearly demonstrates that the model understands decipherable BIJECTIVE maps better than the undecipherable NON-BIJECTIVE maps. This provides evidence for **task learning** capabilities of LLMs.

Model → Dataset (shuffle rate) ↓	Cipher	20-shot LLaMA	Qwen	Olmo	Gemma
SST-2 ($r = 0.6$)	NON-BIJECTIVE	55.7	65.7	56.9	59.1
	BIJECTIVE	58.7 (+3.0 ↑)	67.4 (+1.7 ↑)	54.5 (-2.4 ↓)	62.6 (+3.5 ↑)
Amazon ($r = 0.6$)	NON-BIJECTIVE	63.7	71.3	72.2	79.1
	BIJECTIVE	74.9 (+11.2 ↑)	76.8 (+5.5 ↑)	75.9 (+3.7 ↑)	83.4 (+4.4 ↑)
HellaSwag ($r = 0.3$)	NON-BIJECTIVE	28.7	53.0	26.3	32.2
	BIJECTIVE	32.0 (+3.3 ↑)	60.5 (+7.5 ↑)	27.0 (+0.7 ↑)	36.2 (+4.0 ↑)
WinoGrande ($r = 0.1$)	NON-BIJECTIVE	53.7	61.3	53.5	62.7
	BIJECTIVE	55.6 (+1.9 ↑)	63.4 (+2.1 ↑)	54.8 (+1.3 ↑)	63.1 (+0.4 ↑)

Table 1: LLM accuracy (reported in %) with 20-shot demonstrations, under BIJECTIVE and NON-BIJECTIVE cipher. We fix it to a reasonable number here to demonstrate the gap, though later we provide an analysis on the effect of shuffle rate (§6.1). The numbers inside the parenthesis shows the change from NON-BIJECTIVE to BIJECTIVE encoding (gains in green↑ and losses in red↓). In majority of cases, we observe **consistent performance gains under BIJECTIVE cipher**.

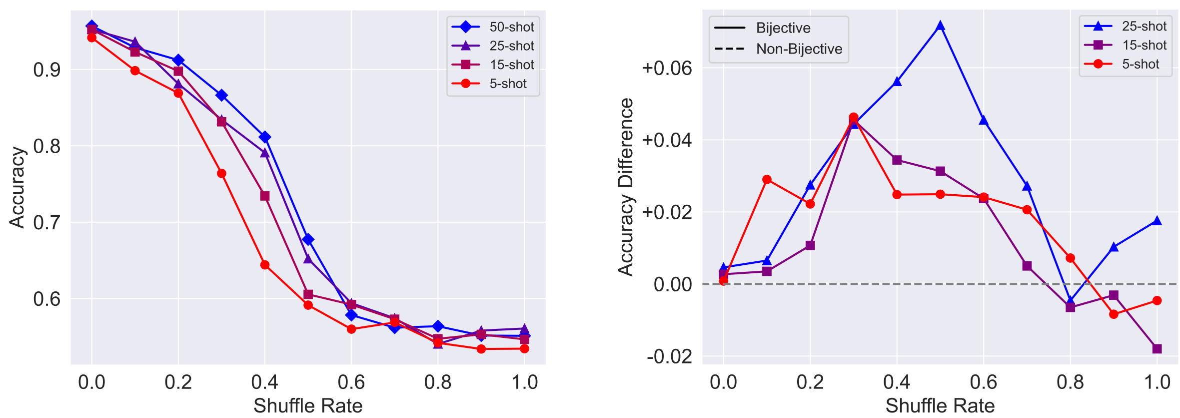


Figure 2: LLaMa 3.1 8B on SST-2 dataset. **Left:** With the BIJECTIVE cipher, accuracy decreases smoothly as the shuffling rate increases, highlighting the difficulty in interpreting the ciphered text. With more demonstrations accuracy also increases, suggesting that the model's ability to solve BIJECTIVE cipher. **Right:** y-axis shows the accuracy gap between BIJECTIVE and NON-BIJECTIVE ciphers. For very high shuffle rates (e.g, > 0.8) tasks become very hard to understand (for the model and even humans) as as the task becomes ill-defined.

6 Further Empirical Analysis

6.1 Effect of Shuffle Rates

Figure 2 illustrates the performance of Llama 3.1 on SST-2 dataset with priority sampling. We can observe a consistent and clear gap between BIJECTIVE ciphers and NON-BIJECTIVE ciphers across a range of shuffle rates, indicating the model's ability to decipher bijections.

6.2 Effect of Number of Demonstrations

In Table 2, we present the gap in performance between BIJECTIVE and NON-BIJECTIVE ciphers under the effect of number of ICL demos. Overall, the BIJECTIVE cipher consistently outperforms the NON-BIJECTIVE cipher across different numbers of demonstrations. Increasing the number of demonstrations generally results in a larger gap between BIJECTIVE and NON-BIJECTIVE ciphers. However, beyond a certain threshold, this effect plateaus, and additional demonstrations have a diminishing impact. Figure 2 (on the right) also shows this visually for SST-2 dataset.

6.3 Effect of Models Size

Figure 3 shows the effect of model size on this gap. As the model size increases, performance for both BIJECTIVE and NON-BIJECTIVE ciphers improve, but the gap between them remains existent. This indicates that decipherability of BIJECTIVE ciphers exists across models of different sizes.

6.4 Probing Representations

To understand the LLMs' internal processing of ciphered inputs, we use Logit Lens (nostalgebraist,

Shots → / Dataset (shuffle rate)↓	Cipher	Model: Llama 3.1 8B					
		5-shot	10-shot	15-shot	20-shot	25-shot	50-shot
SST-2 ($r = 0.6$)	NON-BIJECTIVE	53.6	57.3	56.8	55.7	54.9	56.6
	BIJECTIVE	56.0 (+2.4 ↑)	57.6 (+0.3 ↑)	59.2 (+2.4 ↑)	58.7 (+3.0 ↑)	59.4 (+4.5 ↑)	57.8 (+1.2 ↑)
Amazon ($r = 0.6$)	NON-BIJECTIVE	60.8	67.9	66.5	63.7	65.7	67.0
	BIJECTIVE	65.7 (+4.9 ↑)	74.6 (+6.7 ↑)	75.6 (+9.1 ↑)	74.9 (+11.2 ↑)	74.6 (+8.9 ↑)	77.3 (+10.3 ↑)
HellaSwag ($r = 0.3$)	NON-BIJECTIVE	31.2	29.5	30.0	28.7	29.3	29.3
	BIJECTIVE	33.8 (+2.6 ↑)	33.1 (+3.6 ↑)	32.8 (+2.7 ↑)	32.0 (+3.3 ↑)	32.7 (+3.5 ↑)	32.0 (+2.7 ↑)
WinoGrande ($r = 0.1$)	NON-BIJECTIVE	55.7	54.5	54.8	53.7	52.4	52.0
	BIJECTIVE	55.8 (+0.1 ↑)	57.8 (+3.3 ↑)	55.6 (+0.8 ↑)	55.6 (+1.9 ↑)	55.1 (+2.7 ↑)	54.4 (+2.4 ↑)

Table 2: Task accuracy (reported in %) with varying numbers of ICL examples under BIJECTIVE vs. NON-BIJECTIVE. The numbers inside the parenthesis shows the change from NON-BIJECTIVE to BIJECTIVE encoding. With various number of demonstrations, LLMs get a higher accuracy under Bijective substitution.

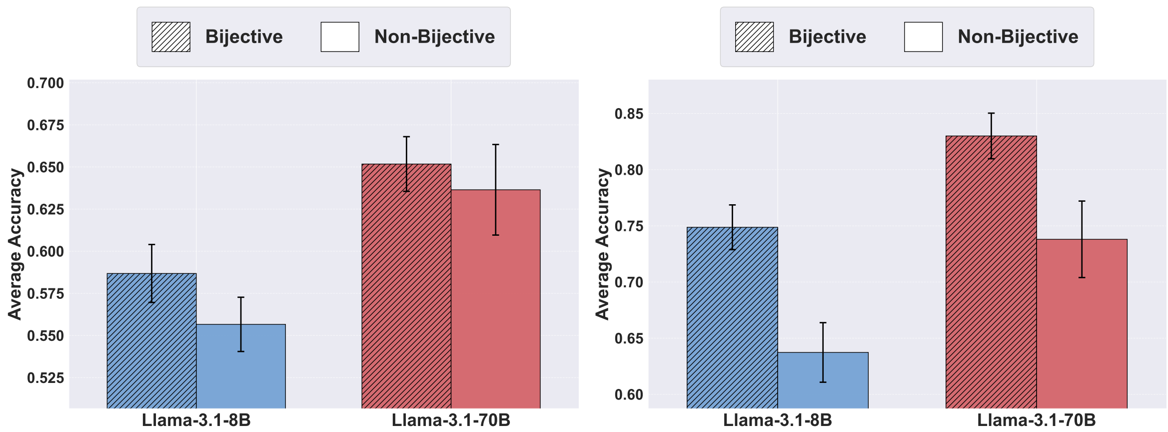


Figure 3: Accuracy comparison of Llama-3.1-8B and Llama-3.1-70B models on SST-2 (left) and Amazon (right) datasets under BIJECTIVE and NON-BIJECTIVE ciphers. The experimental setting is 20-shot with $r = 0.6$. Larger models outperform smaller ones under both ciphers and BIJECTIVE consistently yields higher accuracy.

2020) to probe LLMs's intermediate layer representations. Logit Lens uses embeddings of a token from intermediate layers and uses the final LM head to decode it as the next token. The probing task here is focused on Amazon sentiment dataset and uses Llama 3.1 .

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 correspondence 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):

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

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

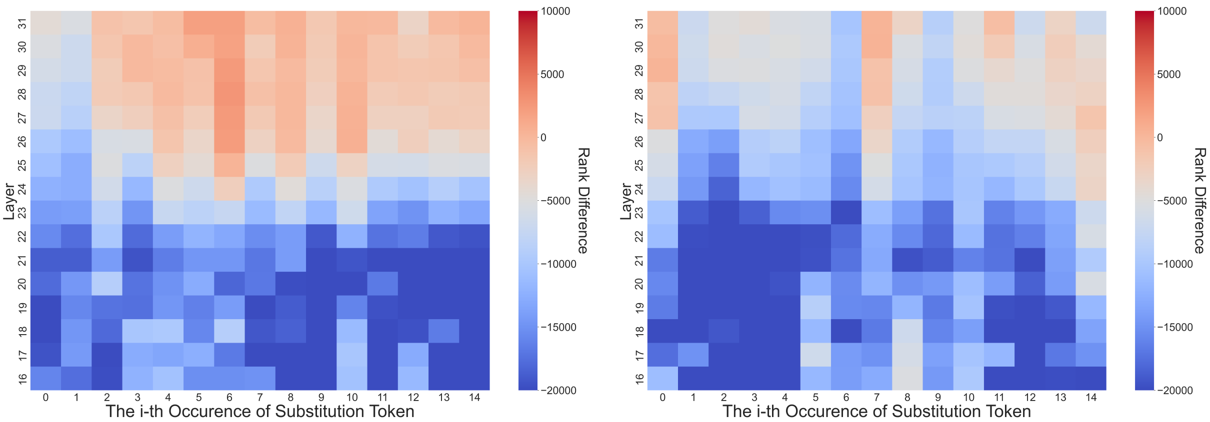


Figure 4: x-axis indicates the i-th occurrence of ciphered tokens in the Llama 3.1 context. y-axis indicates the rank difference (Eq 1). Positive values (red) indicate the model's preference for substituted tokens over original ones. In the Bijective cipher (left), we see a preference that favors substituted tokens. However, there is no clear preference in the Non-Bijective setting (right).

LLM representations favor substituted tokens in Bijective cipher: For Bijective cipher (Fig.4; left) as the model observes more substitutions, the 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.4; right) as there is nothing to decipher.

7 Discussion and Conclusion

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 bijection vs random shuffling and de-emphasize any absolute performance numbers which could have resulted from memorization of the original task.

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 systematic method to quantify learning, distinct from the previous work such as Pan et al. (2023). 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.

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 using ICL CIPHERS 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 16 settings (four models $\times$ four datasets), expanding our study to include more and larger models would strengthen our findings. The largest model we tested was Llama 3.2 70B, due to lack of more compute resources. Additionally, we did not evaluate 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.

More 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).

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 Experimental Details

A.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 a underline.

A.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.

A.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 §4.1.

A.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 develop set as the test set, which contains 1267 examples. Its xl training set is used as demo pool, which has 40k examples.

A.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.

B 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 spaǒ0f2 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

C 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.

D Priority vs Non-Priority Sampling

Figure 5 shows peformance of LLaMa 3.1 8B on SST-2 dataset with non-priority sampling. Comparing with Figure 2, they demonstrate similar trend but there are more variance in non-priority sampling due to the its random nature. Therefore, we use priority sampling throughout our experiments for more steady results.

Figure 7 shows accuracy comparison of Llama-3.1-8B and Llama-3.1-70B on SST-2 and Amazon datasets with non-priority sampling under BIJECTIVE and NON-BIJECTIVE substitution strategies.

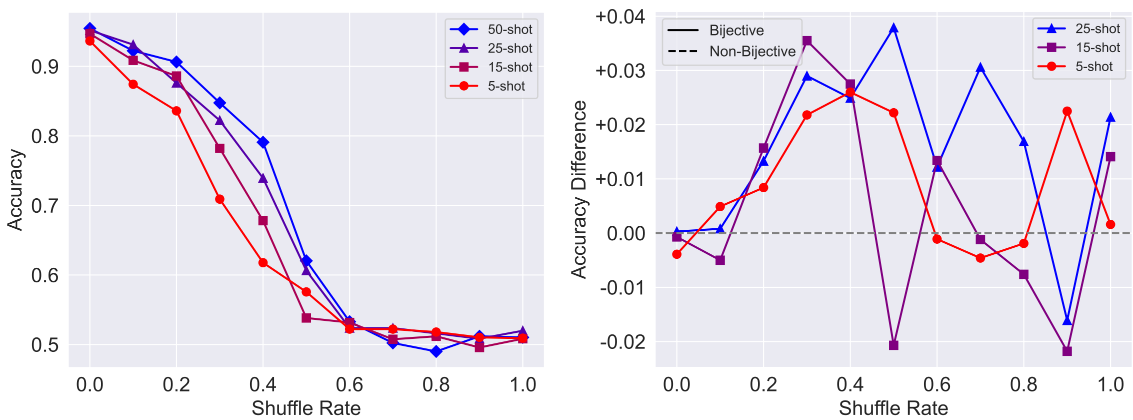


Figure 5: **Left:** Peformance of LLaMa 3.1 8B on SST-2 dataset with non-priority sampling. **Right:** The y-axis displays the accuracy gap between BIJECTIVE and NON-BIJECTIVE ciphers.

E 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. Table 4 shows that the gap between BIJECTIVE and NON-BIJECTIVE ciphers moderately increases for noun-constrained shuffling. This means that the model is more effectively learning to solve ICL ciphers.

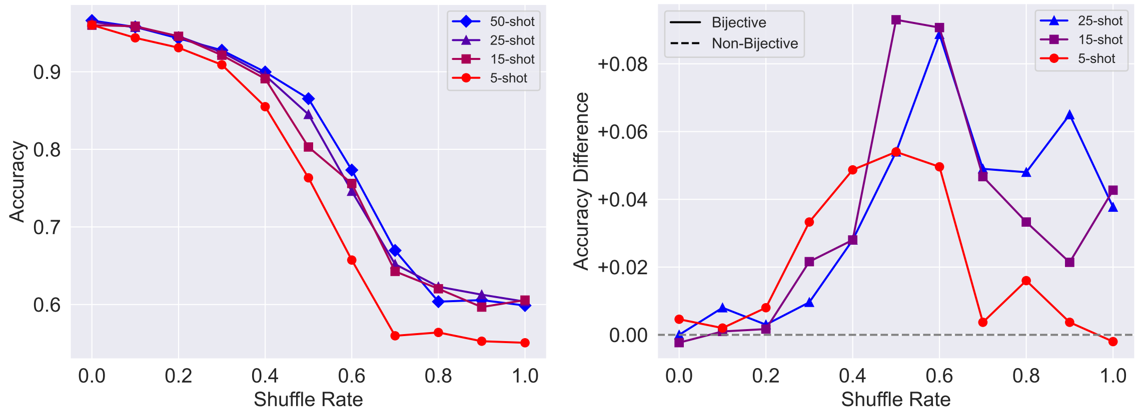


Figure 6: LLama3 accuracy on Amazon dataset with priority sampling (similar to Fig.2). The left plot shows the accuracy change of BIJECTIVE cipher. The right plot shows the gap between BIJECTIVE and NON-BIJECTIVE cipher.

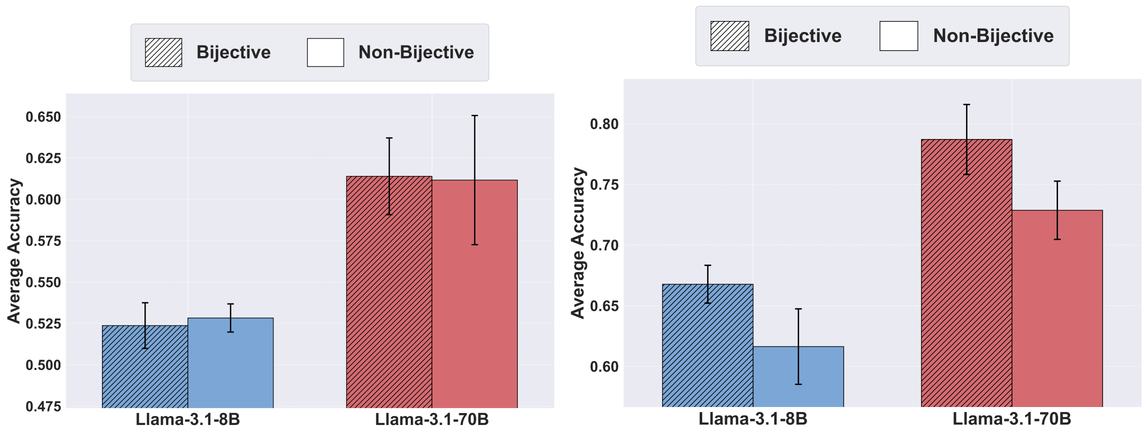


Figure 7: Accuracy comparison of Llama-3.1-8B and Llama-3.1-70B models on SST-2 (**Left**) and Amazon ((**Left**)) datasets (non-priority sampling) under BIJECTIVE and NON-BIJECTIVE substitution strategies.

F 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 Figure 9. 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.

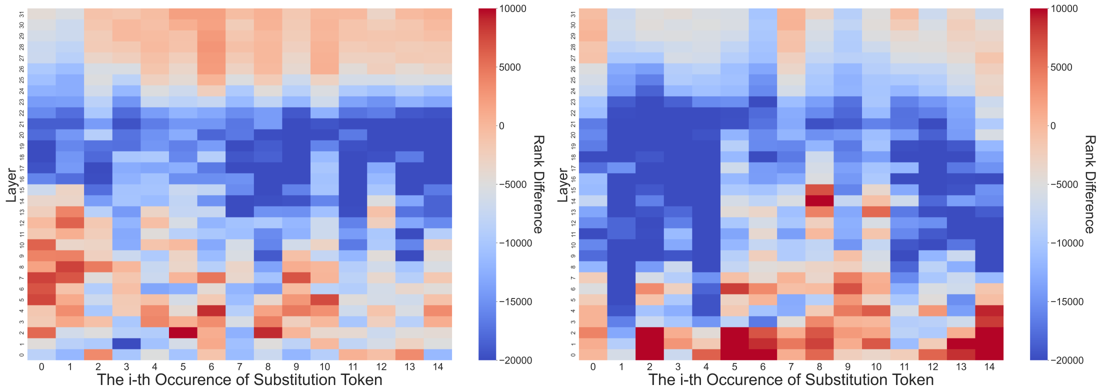


Figure 8: Whole heatmap of original token rank minus substitution token rank on Amazon. **Left:** BIJECTIVE substitution **Right:** NON-BIJECTIVE substitution

Dataset ↓	Cipher	LLaMA 20-shot All	LLaMA 20-shot Noun
HellaSwag	NON-BIJECTIVE	28.7	31.1
	BIJECTIVE	32.0 (+3.3 ↑)	35.5 (+4.4 ↑)
WinoGrande	RANDOM	53.7	54.6
	BIJECTIVE	55.6 (+1.9 ↑)	56.7 (+2.1 ↑)

Table 4: LLM accuracy (reported in %) with 20-shot demonstrations, under BIJECTIVE and RANDOM cipher with zipfian shuffling and informative sampling. The shuffle rates for HellaSwag and WinoGrande are 0.3 and 0.1 respectively. "All" operates shuffling on all the tokens while "Noun" constrains shuffling to only nouns.

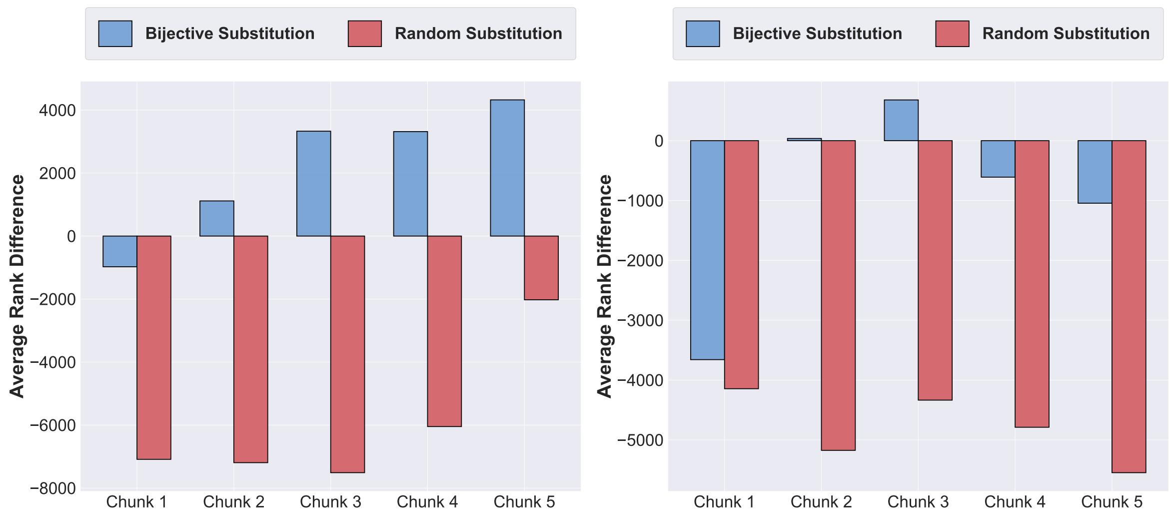


Figure 9: Average rank differences (original token rank - substitution token rank) in SST-2 (left) and Amazon (right) datasets for BIJECTIVE (blue) and NON-BIJECTIVE (red) Cipher strategies over 15 occurrences, divided into 5 chunks of size 3. Rank difference serves as a proxy for the model's deciphering ability. Under BIJECTIVE substitution, this ability improves with more exposure to substituted tokens, while NON-BIJECTIVE substitution shows no clear pattern.