From Unstructured Data to In-Context Learning: Exploring What Tasks Can Be Learned and When

Kevin Christian Wibisono
University of Michigan, Statistics
kwib@umich.edu

Yixin Wang
University of Michigan, Statistics
yixinw@umich.edu

Abstract

Large language models (LLMs) like transformers demonstrate impressive in-context learning (ICL) capabilities, allowing them to make predictions for new tasks based on prompt exemplars without parameter updates. While existing ICL theories often assume structured training data resembling ICL tasks (e.g., x-y pairs for linear regression), LLMs are typically trained unsupervised on unstructured text, such as web content, which lacks clear parallels to tasks like word analogy. To address this gap, we examine what enables ICL in models trained on unstructured data, focusing on critical sequence model requirements and training data structure. We find that many ICL capabilities can emerge simply from co-occurrence of semantically related word pairs in unstructured data; word analogy completion, for example, can provably arise purely through co-occurrence modeling, using classical language models like continuous bag of words (CBOW), without needing positional information or attention mechanisms. However, positional information becomes crucial for logic reasoning tasks requiring generalization to unseen tokens. Finally, we identify two cases where ICL fails: one in logic reasoning tasks that require generalizing to new, unseen patterns, and another in analogy completion where relevant word pairs appear only in fixed training positions. These findings suggest that LLMs' ICL abilities depend heavily on the structural elements within their training data.[1]

1 Introduction

Large language models (LLMs) such as transformers demonstrate remarkable in-context learning (ICL) abilities [10]: without any parameter updates, they can recognize tasks and generate predictions from prompt examples. For instance, given the prompt *dog anjing, cat kucing, lion singa, elephant*, a well-trained LLM should detect the English-to-Indonesian pattern in the prompt and predict *gajah*—the Indonesian translation for *elephant*—as the most likely next token. The ICL capabilities of LLMs are surprising for two main reasons. First, these models are trained in an unsupervised manner on unstructured natural language data through next-token prediction, without any loss function specifically designed for ICL. Second, the training data for LLMs likely lacks sequences resembling typical ICL prompts, i.e., of the form $c_1 d_1 \cdots c_K d_K$, where (c_k, d_k)'s represent word pairs with specific semantic relationships.

Many efforts have sought to understand ICL from theoretical and empirical perspectives, e.g., gradient descent in regression and Bayesian inference. While insightful, these analyses often rely on structured training data that mirrors ICL tasks. For instance, they train on sequences of x-y pairs from various linear regression tasks and test on similar data. In practice, however, LLMs are trained in an unsupervised manner on unstructured text data, such as web content, which bears little resemblance

[1] Software that replicates the empirical studies can be found at `https://github.com/yixinw-lab/icl-unstructured`. Details on implementation, experiments and data sets are provided in Appendix K.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

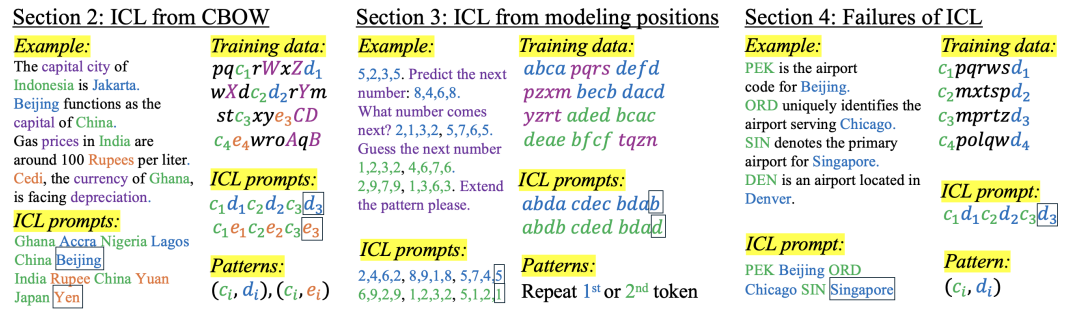

Figure 1: This paper identifies essential components for in-context learning (ICL) from pre-training on unstructured natural language data. Left sub-panels, right sub-panels, and boxed letters denote NLP examples, our abstractions, and expected outputs, respectively. Section 2 shows that ICL for word analogy tasks can arise via modeling co-occurrence information using classical language models like continuous bag of words (CBOW) (violet represents relationship-specific nuisance tokens). Section 3 establishes the necessity of modeling positional information and blocked nuisance structure for ICL tasks, enabling pattern recognition and generalization to novel tokens (violet represents nuisance tokens). Section 4 presents scenarios where ICL fails, providing theoretical explanations that underscore the critical role of training data structure in enabling ICL in language models.

to typical ICL tasks like word analogy. Consequently, these analyses may only partially capture the complexities of ICL.

This work. We investigate common ICL tasks to identify what tasks can be learned in context by a model trained on unstructured data. Specifically, we examine essential components of sequence modeling that enable in-context learning, along with requirements on the unstructured training data.

The first set of (theoretical and empirical) results focuses on ICL for word analogy completion using frequently co-occurring tokens [10, 50]. This task involves identifying relationships between word pairs, such as *(country)-(capital)* and *(English word)-(Indonesian translation)*, then applying the same relationship to complete a sequence. For this task (see left of Figure 1), we explore cases where training sentences contain one or two types of word pairs with distinct semantic relationships. We prove that, in most cases, ICL can arise by simply modeling word co-occurrence using classical (pre-transformer) language models like continuous bag of words (CBOW) [36], without needing positional information or attention mechanisms.

The second set of results involves ICL for logic reasoning tasks that require recognizing patterns that do not commonly co-occur in a sentence, such as *(word)-(first letter)* [13, 62]. For this task (see middle of Figure 1), we investigate scenarios where training sentences contain one or two distinct patterns, as well as a more realistic scenario where nuisance tokens are present. We prove that *positional information and blocked nuisance structure* (e.g., *pqrs* in Figure 1) *are crucial for the success of ICL in these tasks*. This finding aligns with Chen et al.'s [2024] observation that parallel structures in pre-training data support ICL. We also find that learned positional embeddings generally perform better, except in scenarios where the nuisance tokens are not clustered in blocks.

Finally, we present two scenarios where ICL fails regardless of model architectures (see right of Figure 1). In the first scenario (left example), we consider a logic reasoning task that involves identifying and completing meta-patterns within sequences. Here, each training sequence repeats the pattern established by its starting tokens; the ICL task sequence then requires the model to recognize this meta-pattern of repetition and generalize it to a novel, unseen starting pattern. In the second scenario (right example), we examine a word analogy completion task in which relevant word pairs appear in the unstructured training sentences but are restricted to specific fixed positions. These findings, along with their empirical and theoretical explanations, underscore that *LLMs require specific structures in the pre-training data to exhibit ICL ability*.

Summary of contributions. We (1) theoretically and empirically show that ICL for word analogy tasks with semantically related word pairs can arise from modeling *co-occurrence patterns* using CBOW, (2) prove that, to recognize token patterns and generalize them to novel tokens, ICL requires *modeling positional information* and *blocked nuisance structure*, and (3) present scenarios where ICL fails, highlighting the crucial role of *training data structure* for ICL.

Related work. Below, we highlight some of these studies and explain how our research aligns with, yet differs from, these approaches. We include a detailed discussion of related work in Appendix A. Numerous studies have connected ICL to classical methods, including gradient descent [3, 5, 17, 55, 66], Bayesian inference [14, 56, 67], and Newton's method [20]. In contrast, *our work links ICL to the continuous bag of words (CBOW) model*, showing that ICL for word analogy tasks can be achieved by learning co-occurrence patterns. Several studies have examined the pre-training aspects of ICL, such as data distribution [11, 30, 37] and task diversity [42, 63]. By comparison, *our work emphasizes the importance of co-occurrence, positional information, and training data structure* for ICL to arise. Other research has explored ICL in specific data-generating processes, such as discrete functions [8] and autoregressive processes [45]. In contrast, *our work centers on data characterized by semantically related word pairs and repeating token patterns*.

2 In-context learning can arise by modeling co-occurrence via CBOW

In this section, we focus on in-context learning (ICL) for word analogy tasks involving word pairs that frequently co-occur in training sentences; see Figure 1 (left). To motivate the discussion, we present two experiments using the LLaMA 2 model [51] involving countries (or US states) and their capital cities (see Appendix K for data sources). The prompts follow the format $c_1 d_1, c_2 d_2, \cdots, c_6 d_6, c_7$, where c_i is a country (or US state) and d_i is its capital city. In this scenario, we consider ICL successful if the model outputs d_7—the capital city of c_7—as the most likely token.

Experiment 1. We consider all 160 countries with a population exceeding one million in 2022. Among these countries, 31 have capital cities that are not their most populous cities, denoted by *type A*. The remaining 129 countries fall under *type B*. Each ICL prompt includes three type A countries among $c_1, \cdots, c_6$ to emphasize that the desired relationship is *(country)-(capital)* rather than *(country)-(largest city)*. Subsequently, we randomly generate 1,000 prompts, with 500 having a c_7 being a type A country and 500 having a c_7 being a type B country. The model's ICL accuracy is 0.58 for type A and 0.96 for type B.

Experiment 2. We consider all 50 US states, among which 33 are of *type A* and 17 are of *type B*, similarly defined. Following the setup in Experiment 1, we generate prompts for these states. The ICL accuracy is 0.69 for type A and 0.84 for type B.

In both experiments, LLaMA 2 performs better on type B prompts (i.e., the capital city as the largest city). Since larger cities tend to appear more frequently as compared to smaller ones in the model's pre-training data, this naturally raises the question: *Can/does ICL with frequently co-occuring word pairs arise purely from modeling co-occurrence patterns?*

ICL via classical non-transformer-based language models. We prove that, for word analogy tasks with frequently co-occurring word pairs, ICL can be achieved by modeling token co-occurrence—without needing positional encoding or attention mechanisms—using classical, non-transformer language models such as the continuous bag of words (CBOW) model [36]. (It *does not* imply that ICL in transformer-based models arises through learning co-occurrence patterns.) We utilize CBOW variant where each center word is modeled conditional on all other words in a sentence, not just neighboring words. Specifically, each word w has center and context embeddings u_w and v_w of the same dimension. Given a sentence $x_1 x_2 \cdots x_I$, the i-th word (x_i) is distributed as $p(x_i = k \mid x_{-i}) \propto \exp((u_k^\top \sum_{j \neq i} v_{x_j})/(I-1))$, with u_w's and v_w's learned by minimizing cross-entropy losses across all positions.

Roadmap of Section 2. In Section 2.1, we begin by considering a simple ICL task of the form $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$, where (c_i, d_i) represents a frequently co-occuring word pair (e.g., a country and its capital city) and $i_1, i_2, \cdots, i_{\ell+1}$ are all distinct. The focus is to investigate whether a trained CBOW model can correctly output d_{i_ℓ}. We also explore two other scenarios: ICL tasks of the form $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$ and $c_{i_1} e_{i_1} \cdots c_{i_\ell} e_{i_\ell} c_{i_{\ell+1}}$ in Section 2.2 (two connected word relationships), as well as $c_{i_1} d_{i_1} \cdots c_{i_{\ell-1}} d_{i_{\ell-1}} c_{i_\ell}$ and $e_{i_1} f_{i_1} \cdots e_{i_\ell} f_{i_\ell} e_{i_{\ell+1}}$ (two disjoint word relationships) in Section 2.3. Section 2.4 concludes with synthetic experiments supporting the theory.

2.1 In-context learning on single-relationship word analogy tasks

We investigate ICL in single-relationship word analogy tasks, where the training data contains only one type of relationship between frequently co-occurring word pairs. This task takes the form of $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$, where (c_i, d_i) pair represents a frequent co-occurrence, such as a country and its capital city. The vocabulary consists of $c_{1:K}, d_{1:K}, r_{1:L}$, where $r_i's$ represent other words (e.g.,

stop words). We first introduce Theorem 1, which states that ICL can arise if each sentence consists of exactly one (c_i, d_i) pair, as long as the number of in-context examples (ℓ) is not too large. To simplify calculations, we replace the cross-entropy loss with squared loss by removing the softmax activation and comparing outputs against the one-hot encoding of target words. The proof is in Appendix B.

Theorem 1 (ICL on single-relationship word analogy tasks)**.** *Let $K, L \geq S \geq 3$. Suppose each training sentence of length S is generated by selecting one (c_i, d_i) pair and $S-2$ distinct r_i's uniformly at random. We train a CBOW model with the squared loss and a sufficiently large embedding dimension on these sentences. Given a prompt $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$ with distinct i_k's, the model correctly predicts $d_{i_{\ell+1}}$ if and only if $2\ell + 1 < \frac{KL(S-1)^3}{(K+L)(S-2)^2(S-1)+K(S-2)(S-1)^2-2(S-2)^4}$.*

As an example, when each training sentence contains exactly one *country-capital* pair (i.e., (c_i, d_i)), Theorem 1 says that a trained CBOW model will correctly predict $d_{i_{\ell+1}}$ (i.e., the capital city of $c_{i_{\ell+1}}$) given an ICL prompt of the form $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$, provided that the prompt length is not too large. Intuitively, this behavior is due to the presence of $c_{i_{\ell+1}}$ in the ICL prompt, leading the model to correctly predict $d_{i_{\ell+1}}$ given the frequent occurrences of the pair $(c_{i_{\ell+1}}, d_{i_{\ell+1}})$ in the training data. However, when the prompt length is too large, the model will instead predict one of the r_i's (see Theorem 1's proof in Appendix B for more details). Moreover, if we let $L \to \infty$ and fix K and S, the condition in Theorem 1 becomes $2\ell + 1 < K(S-1)^2/(S-2)^2$. This inequality trivially holds if the prompt length is set to be $S-1$ to match the length of the training sentences.

Furthermore, it is possible to adapt the proof of Theorem 1 to handle the case when each sentence comprises exactly two (not one) different (c_i, d_i) pairs. In this case, letting $L \to \infty$ and fixing K and S, the model correctly predicts $d_{i_{\ell+1}}$ given the same ICL prompt if and only if $2\ell + 1 < \frac{K(K-2)(S-1)^2}{(K-2)(S-2)(S-4)-K}$. This upper bound is strictly larger than $K(S-1)^2/(S-2)^2$: when each sentence contains exactly two (c_i, d_i) pairs, ICL under the squared loss holds for longer prompts.

Experiments. To empirically verify Theorem 1 and its generalizations, we conduct experiments using the cross-entropy loss with $S = 8$, $K = 10$, $L = 20$, and $\ell = 3$. We explore multiple (p_0, p_1, p_2) values, where p_k denotes the probability of having exactly k pairs of (c_i, d_i) in the sentence. For each (p_0, p_1, p_2) triple, we also introduce a more realistic setting where c_i and d_i do not always appear together by considering its *corrupted* version. In this setup, each (c_i, d_i) pair has a 25% chance of being replaced with (c_i, r_j) and a 25% chance of being replaced with (d_i, r_j) for some $j \in [L]$. More details are provided in Appendix K.

Results. Table 1 displays the average accuracy for each scenario, calculated over 10 repetitions. Notably, when (p_0, p_1, p_2) is $(0, 1, 0)$ or $(0, 0, 1)$, ICL under the cross-entropy loss achieves zero accuracy, in contrast to perfect accuracy with the squared loss as shown in Theorem 1. We believe this difference in accuracy is an artifact of the loss functions used, although its relevance is limited by the fact that it is unlikely for every sentence to contain at least one (c_i, d_i) pair, in reality. On the other hand, perfect ICL performance is observed in other settings (e.g., when the training sentences contain either zero, one, or two (c_i, d_i) pairs) in both the clean and corrupted scenarios. For an in-depth comparison of ICL performance using both the squared and cross-entropy loss across various numbers of demonstration examples, see Appendix C.

Table 1: ICL on different single-relationship word analogy tasks, averaged over 10 repetitions, demonstrates stable, good performance across embedding dimensions (d_E), as Theorem 1 suggests. The corrupted setting also demonstrates excellent ICL ability under certain scenarios.

	Clean		Corrupted	
(p_0, p_1, p_2)	$d_E = 10$	$d_E = 100$	$d_E = 10$	$d_E = 100$
$(0, 1, 0)$	0	0	0	0
$(0, 0, 1)$	0	0	0	0
$(1/2, 1/2, 0)$	1	0.99	0	0
$(1/2, 0, 1/2)$	1	1	1	1
$(0, 1/2, 1/2)$	1	1	0	0.01
$(1/3, 1/3, 1/3)$	1	1	1	1

2.2 In-context learning on dual-connected-relationship word analogy tasks

Building on the scenario that contains only a single type of relationship between frequently co-occurring word pairs, namely (c_i, d_i), we now explore ICL on dual-connected-relationship word

Table 2: ICL on dual-*connected*-relationship tasks, averaged over 10 repetitions, achieves perfect accuracy when $(p_0, p_1, p_2) \in \{(1/2, 0/1, 2), (0, 1/2, 1/2), (1/3, 1/3, 1/3)\}$ regardless of architectures and embedding dimensions (d_E), as Theorem 2 suggests. When $(p_0, p_1, p_2) = (1/2, 1/2, 0)$, ICL performs better under imbalanced or extreme scenarios and with larger d_E.

	Balanced		Imbalanced		Extreme	
(p_0, p_1, p_2)	d_E = 10	d_E = 100	d_E = 10	d_E = 100	d_E = 10	d_E = 100
$(0, 1, 0)$	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)
$(0, 0, 1)$	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0.07, 0.10)	(0, 0)
$(1/2, 1/2, 0)$	(0.53, 0.47)	(0.51, 0.50)	(0.69, 0.68)	(1, 1)	(0.94, 0.93)	(1, 1)
$(1/2, 0, 1/2)$	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
$(0, 1/2, 1/2)$	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
$(1/3, 1/3, 1/3)$	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)

analogy tasks. Here, some words frequently co-occur with two different types of words in the training data, represented by the relationships (c_i, d_i) and (c_i, e_i). For instance, c_i might represent a country, d_i its capital city, and e_i its currency. The vocabulary is comprised of $c_{1:K}, d_{1:K}, e_{1:K}, r_{1:L}$, where r_i's represent other words. Moreover, the corresponding ICL tasks take the form $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$ and $c_{i_1} e_{i_1} \cdots c_{i_\ell} e_{i_\ell} c_{i_{\ell+1}}$, where the model is expected to output $d_{i_{\ell+1}}$ and $e_{i_{\ell+1}}$, respectively. These can be regarded as *task selection* since the model should use the in-context examples to infer the tasks. We present Theorem 2, stating that a trained CBOW model can perform task selection if each sentence contains exactly two distinct (c_i, d_i) or two distinct (c_i, e_i) pairs with uniform probability.[2]

Theorem 2 (Task selection in CBOW)**.** *Let $K, L \geq 2$ and $S \geq 5$. Suppose each training sentence of length S is generated by selecting two distinct (c_i, d_i) pairs or (c_i, e_i) pairs, and $S - 4$ distinct r_i's uniformly at random. We train a CBOW model with the squared loss and a large enough embedding dimension. Given a prompt $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$ ($c_{i_1} e_{i_1} \cdots c_{i_\ell} e_{i_\ell} c_{i_{\ell+1}}$) with distinct i_k's, the model is more likely to predict $d_{i_{\ell+1}}$ ($e_{i_{\ell+1}}$) than $e_{i_{\ell+1}}$ ($d_{i_{\ell+1}}$). (The proof is in Appendix D.)*

According to Theorem 2, when each training sentence includes two (c_i, d_i) pairs or two (c_i, e_i) pairs, a trained CBOW model is capable of performing task selection. To intuitively understand this result, consider the ICL prompt of the first type, i.e., $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$. Here, the output is more likely to be $d_{i_{\ell+1}}$ than $e_{i_{\ell+1}}$ since $d_{i_{\ell+1}}$ co-occurs with the other d_{i_j}'s in the training data (and $e_{i_{\ell+1}}$ does not). Note that in Theorem 2, we unrealistically require each sentence to contain either two distinct (c_i, d_i) pairs or (c_i, e_i) pairs. However, this condition is not necessary as we empirically show next.

Experiments. We use the cross-entropy loss with $S = 8$, $K = 10$, $L = 60$, and $\ell = 3$. Each training sentence is equally likely to be a *cd* sentence (i.e., containing (c_i, d_i) pairs) or a *ce* sentence (i.e., containing (c_i, e_i) pairs), but not both. We explore multiple (p_0, p_1, p_2)'s, where p_k is the probability of having exactly k pairs of (c_i, d_i) for a *cd* sentence, or k pairs of (c_i, e_i) for a *ce* sentence. Additionally, we introduce three different scenarios: *balanced*, where all L random words are equally likely to occur in both *cd* and *ce* sentences; *imbalanced*, where $L/3$ words are more likely to occur in *cd* (*ce*) sentences; and *extreme*, where $L/3$ of the words can only occur in *cd* (*ce*) sentences. More details are provided in Appendix K.

Results. Table 2 shows the accuracies of both tasks for each scenario, averaged over 10 repetitions. We observe a perfect accuracy when $(p_0, p_1, p_2) \in \{(1/2, 0/1, 2), (0, 1/2, 1/2), (1/3, 1/3, 1/3)\}$ across all embedding dimensions and scenario types. The near-zero accuracy when (p_0, p_1, p_2) or $(0, 1, 0)$ or $(0, 0, 1)$ is again an artifact of the cross-entropy loss, as discussed in Section 2.1.

Interestingly, ICL works in the imbalanced and extreme scenarios when $(p_0, p_1, p_2) = (1/2, 1/2, 0)$, where sentences do not contain more than one (c_i, d_i) or (c_i, e_i) pair. To see this, consider the balanced scenario where each r_i is equally probable to appear in both types of sentences. Given a prompt of the form $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$, it is easy to see that the model should output $d_{i_{\ell+1}}$ or $e_{i_{\ell+1}}$ with equal probability. On the other hand, in the imbalanced and extreme scenarios, the information from the r_i's can allow for task selection, thus contributing to the success of ICL.

[2]We can also theoretically show that ICL works (up to a certain number of training examples) in this scenario, but the calculations are extremely tedious. Therefore, we only present empirical evidence in Table 2.

Table 3: ICL on dual-*disjoint*-relationship tasks, averaged over 10 repetitions, achieves perfect accuracy when $(p_0, p_1, p_2) \in \{(1/2, 0/1, 2), (0, 1/2, 1/2), (1/3, 1/3, 1/3)\}$ regardless of architectures and embedding dimensions (d_E). When $(p_0, p_1, p_2) = (1/2, 1/2, 0)$, ICL already performs well under the balanced scenario.

	Balanced		Imbalanced		Extreme	
(p_0, p_1, p_2)	d_E = 10	d_E = 100	d_E = 10	d_E = 100	d_E = 10	d_E = 100
$(0, 1, 0)$	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)
$(0, 0, 1)$	(0, 0)	(0, 0)	(0.16, 0.14)	(0, 0)	(0.21, 0.29)	(0, 0)
$(1/2, 1/2, 0)$	(1, 1)	(0.82, 0.83)	(0.28, 0.27)	(0.95, 0.95)	(0.83, 0.85)	(0.91, 0.91)
$(1/2, 0, 1/2)$	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
$(0, 1/2, 1/2)$	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
$(1/3, 1/3, 1/3)$	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)	(1, 1)

2.3 In-context learning on dual-disjoint-relationship tasks

We next replicate the experiments in Section 2.2, but with disjoint word pair relationships of two distinct types with no overlapping tokens, i.e., (c_i, d_i) and (e_i, f_i). For example, (c_i, d_i) represents a country and its capital city, and (e_i, f_i) represents a company and its CEO. Our vocabulary consists of $c_{1:K}, d_{1:K}, e_{1:K}, f_{1:K}, r_{1:L}$, where r_i's represent other words; see Appendix K for details.

Results. Table 3 summarizes the accuracies of the ICL tasks $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$ and $e_{i_1} f_{i_1} \cdots e_{i_\ell} f_{i_\ell} e_{i_{\ell+1}}$ for each scenario, averaged over 10 repetitions. Similar to the connected setting in Section 2.2, we observe a perfect accuracy when $(p_0, p_1, p_2) \in \{(1/2, 0/1, 2), (0, 1/2, 1/2), (1/3, 1/3, 1/3)\}$ across all embedding dimensions and scenario types. However, when $(p_0, p_1, p_2) = (1/2, 1/2, 0)$, ICL already works well in the balanced scenario. Intuitively, this is because the two relationships are disjoint, thus making task selection easier.

In addition, we consider a *contaminated* version of the training data where *cd* (*ef*) sentences can contain some e_i's and f_i's (c_i's and d_i's). We also obtain a perfect accuracy when $(p_0, p_1, p_2) \in \{(1/2, 0/1, 2), (0, 1/2, 1/2), (1/3, 1/3, 1/3)\}$ across all embedding dimensions and scenario types.

2.4 Experiments on a synthetic corpus

We conduct experiments on a synthetic corpus consisting of *(country)-(capital)* and *(country)-(IOC code)* relationships. Each sentence in the corpus is categorized into exactly one of six possible categories: (1) exactly one country-capital pair; (2) exactly two country-capital pairs; (3) exactly one country-IOC pair; (4) exactly two country-IOC pairs; (5) exactly one country without any pair; and (6) no country. In sentences with country-capital pairs, each capital city can appear in any position relative to the country. Conversely, in sentences with country-IOC pairs, each IOC code must directly follow the country. The data source and corpus generation process are detailed in Appendix K.

Two models are trained on this corpus: a CBOW and a five-layer two-head autoregressive transformer. Both models have an embedding dimension of 100. We then compare the ICL accuracies for both relationships given one to five in-context examples. For the CBOW model, the country-capital accuracies are $(0.81, 0.82, 0.78, 0.73, 0.65)$ and the country-IOC accuracies are $(0.15, 0.38, 0.59, 0.71, 0.79)$. Here, the i-th number corresponds to the accuracy given i in-context examples. For the transformer, the accuracies are $(0.00, 0.15, 0.34, 0.22, 0.07)$ and $(1.00, 0.77, 0.78, 0.97, 0.99)$, respectively.

When using the transformer, we find that the accuracies for the country-IOC task are significantly higher compared to those for the country-capital task. This is likely because each IOC code consistently follows the corresponding country in the corpus, similar to ICL prompts. On the other hand, ICL fails to work on the country-capital task, where there is no consistent pattern in how each pair occurs in the corpus. Meanwhile, *ICL works decently well on both tasks under the CBOW model*.

3 The essential role of positional information in enabling in-context learning

We examine another common example of in-context learning (ICL), where the task involves predicting the first (or second) token in a sequence. This task resembles general logic reasoning tasks that require recognizing patterns that do not typically co-occur in a sentence, such as *(word)-(first letter)* [13, 62]. While Section 2 shows that positional encoding is irrelevant for ICL in word analogy tasks, positional

information proves essential for such logic reasoning tasks. Specifically, we consider a simpler task of modeling $x_{i_1}x_{i_2}x_{i_3}x_{i_1}$. Theorem 3 underscores the importance of positional information to correctly predict x_{i_1} from $x_{i_1}x_{i_2}x_{i_3}$ in a single-layer model, and provides a construction of an attention-based model achieving zero loss and perfect accuracy on this task. Its proof is in Appendix E.

Theorem 3 (Necessity of modeling positions)**.** *Let the vocabulary be $\mathcal{V} = \{1, 2, \cdots, |V|\}$ and the training sequences take the form $x_{i_1}x_{i_2}x_{i_3}x_{i_1}$, where $x_{i_1} \neq x_{i_2} \neq x_{i_3} \neq x_{i_1}$ are chosen uniformly at random from $\mathcal{V}$. Consider a one-layer model that predicts the last x_{i_1} via a learned function $f(\{x_{i_1}, x_{i_2}\}, x_{i_3})$ using the cross-entropy loss. In this case, it is not possible to achieve pefect accuracy or zero loss. On the other hand, we can achieve zero loss (and thus perfect accuracy) by incorporating positional information, i.e., via a learned function $\tilde{f}(\{(x_{i_1}, 1), (x_{i_2}, 2)\}, (x_{i_3}, 3))$.*

Here, $f(\{x_{i_1}, x_{i_2}\}, x_{i_3})$ represents a scenario where the model lacks positional information (e.g., f is a one-layer autoregressive transformer without positional embeddings). Note that the output of this function is identical for inputs $x_{i_1}x_{i_2}x_{i_3}$ and $x_{i_2}x_{i_1}x_{i_3}$, which leads to the impossibility of attaining zero loss. In contrast, $\tilde{f}(\{(x_{i_1}, 1), (x_{i_2}, 2)\}, (x_{i_3}, 3))$ refers to a scenario where the model has access to positional information. We provide a construction of $\tilde{f}$ that achieves zero loss in Appendix E.

Experiments. We validate Theorem 3 by training transformers with causal masking to autoregressively learn sequences of the form $x_{i_1}x_{i_2}x_{i_3}x_{i_1}$, and assessing their accuracy in predicting the last token on a separate test data of the same pattern. We use $|V| = 20$ and an embedding dimension of 10. We consider these settings: (i) *number of layers*: 1, 5; (ii) *positional embeddings*: learned, sinusoidal, no positional embeddings; and (iii) *train-test split*: each token in the vocabulary is the first token in both the training and test sets (*Both*), each token in the vocabulary is the first token in either set, but not both (*Either*). More details are provided in Appendix K.

Results. Table 4 summarizes the results. Two main findings emerge: (1) for the model to generalize to unseen sentences, each token in $\mathcal{V}$ should be present as the first token in both the training and test sets; (2) positional embeddings are crucial when using only one attention layer. Note that in practice, the condition in (1) is likely met due to the vast size of LLMs' pre-training data.

Table 4: Prediction accuracy with single/multi-layer models. For ICL to occur, the first tokens of training sentences should cover the entire vocabulary (*Both*). Also, positional embeddings are essential, especially in one-layer models.

	Both		Either	
Pos. emb.	1-layer	5-layer	1-layer	5-layer
Learned	1	1	0	0
Sinusoidal	1	1	0	0
No pos. emb.	0.30	0.89	0	0

Multiple layers. Proposition 1 shows that multi-layer models can encode positional information without explicit positional embeddings.

Proposition 1 (Multi-layer models can encode positions)**.** *Consider the sentence $x_{i_1}x_{i_2}x_{i_3}x_{i_1}$. Using a two-layer autoregressive model, the model's final output for predicting the last x_{i_1} is given by $t(x_{i_1}x_{i_2}x_{i_3}) := g_3\left(\{f_1(\{x_{i_1}\}), f_2(\{x_{i_1}\}, x_{i_2})\}, f_3(\{x_{i_1}, x_{i_2}\}, x_{i_3})\right)$ for some f_1, f_2, f_3, and g_3.*

The proof is in Appendix F. Proposition 1 shows that we generally have $t(x_{i_1}x_{i_2}x_{i_3}) \neq t(x_{i_2}x_{i_1}x_{i_3})$, unlike in the one-layer case. Consequently, high accuracy is achievable without positional embeddings, as shown in Table 4. This result parallels findings in Haviv et al. [26] that autoregressive transformers implicitly encode positions, even without positional embeddings.

Roadmap of Section 3. In the rest of this section, we consider settings where each sentence contains repeating patterns. Section 3.1 focuses on a simple scenario where training sentences follow the form *abacdc*, where $a \neq b$ and $c \neq d$, or a noisy variation of it. The ICL prompts maintain the same pattern but use different combinations of *ab* and *cd* from those in the training data. Our goal is to understand what types of training data facilitate ICL in clean or noisy scenarios. Section 3.2 explores a more realistic case where two possible patterns are present: repeating the first letter (*abca*) and repeating the second letter (*abcb*).

3.1 In-context learning on single-pattern tasks

In this section, we examine the case where the training sentences follow the pattern *abacdc*. To replicate real-world training scenarios, we also analyze how incorporating nuisance tokens into the training sentences affects the ICL capability of autoregressive models. To formalize the discussion, let our vocabulary be $\mathcal{V} \cup \mathcal{N}$, where $\mathcal{N}$ represents the nuisance tokens. We define $S = \{(a, b) \mid a, b \in$

Table 5: ICL on single-pattern tasks, averaged over 10 repetitions, achieves near-perfect accuracy in the clean data scenario regardless of architectures and embedding dimension (d_E). The one-noisy scenario is the most challenging, with sinusoidal embeddings giving a higher accuracy. In the block-noisy scenario, learned positional embeddings result in significantly better ICL performance.

	d_E = 10			d_E = 100		
Pos. emb.	Clean	One-noisy	Block-noisy	Clean	One-noisy	Block-noisy
Learned	0.97	0.00	0.95	1.00	0.00	1.00
Sinusoidal	0.66	0.10	0.01	0.96	0.00	0.55
RoPE [48]	0.31	0.00	0.03	0.48	0.00	0.00

$\mathcal{V}, a \neq b\}$ and partition S into S_1 (for training sentences) and S_2 (for ICL prompts). This is to ensure that training sentences are distinct from ICL prompts. Furthermore, we assume $\{c[1] \mid c \in S_1\} = \{c[1] \mid c \in S_2\} = \mathcal{V}$, where $c[i]$ is the i-th element of c. In other words, each token in $\mathcal{V}$ can be the first token in both the training sentences and ICL prompts. We consider three scenarios:

1. *Clean*: Training data follow the form *abacdc* where $ab, cd \in S_1$. ICL prompts follow the form $\underline{abacd}$ where $\underline{ab}$, $\underline{cd} \in S_2$.
2. *One-noisy*: Training data follow the form *abacdc* where $ab, cd \in S_1$, with one nuisance token $n \in \mathcal{N}$ randomly inserted anywhere except the last position (to ensure ICL prompts do not resemble the training data). ICL prompts follow the form $\underline{abacd}$ where $\underline{ab}$, $\underline{cd} \in S_2$.
3. *Block-noisy*: Training data follow the form *abacdc* where $ab, cd \in S_1$, with three consecutive nuisance tokens $n_1, n_2, n_3 \in \mathcal{N}$ randomly inserted while preserving the *aba* and *cdc* blocks. ICL prompts follow the form $\underline{abacdcef}$ where $\underline{ab}$, $\underline{cd}$, $\underline{ef} \in S_2$.

We set the vocabulary size $|V| = 20$, the number of nuisance tokens $N = 20$, and use only one attention layer as additional layers do not improve performance. See Appendix K for more details.

Results. Table 5 reveals interesting phenomena. First, under the clean data scenario, ICL performs exceptionally well, with an observed performance increase with learned positional embeddings and a larger embedding dimension. However, ICL is notably challenging under the one-noisy scenario. In the block-noisy scenario, learned positional embeddings are crucial for satisfactory ICL performance. Theorem 4 formalizes these findings.

Theorem 4 (Blocked nuisance token structure facilitates ICL)**.** *Consider a sufficiently large autoregressive position-aware model that can achieve the minimum possible theoretical loss. Training this model in the one-noisy (block-noisy) scenario results in zero (perfect) ICL accuracy.*

The proof is in Appendix G. Theorem 4 says that ICL works perfectly under the block-noisy scenario, yet fails to work under the one-noisy scenario. However, as shown in Table 5, the use of sinusoidal positional embeddings significantly enhances prediction accuracy in the one-noisy scenario. This may be due to the fact that sinusoidal embeddings can encode relative positional information [53]. For example, training sentences of the form *nabacdc*, where $n \in \mathcal{N}$, may help in predicting the most likely token following the ICL prompt $\underline{abacd}$.

3.2 In-context learning on dual-pattern tasks

We next examine the case where training data and ICL prompts contain two different patterns occurring equally likely: *abcadefd* and *abcbdefe*, where a, b, c and d, e, f are distinct. We consider the *clean* and *block-noisy* scenarios as in Section 3.1, and set $|V| = N = 20$ (details in Appendix K).

Results. Table 6 outlines the ICL performance for both scenario types across different model configurations. Unlike the single-pattern scenario, there is an improvement in performance with five layers compared to one layer, particularly with learned positional embeddings.

This phenomenon is related to the notion of *induction heads*, where at least two layers may be necessary to distinguish the two patterns [38]. This is reflected in Figure 2, which compares the accuracy trajectories of one-layer and five-layer models. While the five-layer setup effectively differentiates the two patterns, the one-layer configuration fails to do so. Meanwhile, in both clean and block-noisy scenarios, learned positional embeddings lead to notably higher accuracies as compared to sinusoidal ones, similar to the single-pattern case.

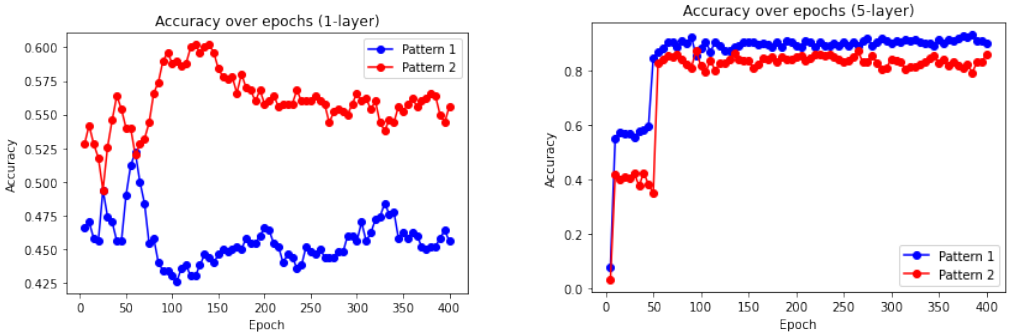

Figure 2: One-layer models fail to differentiate the two patterns in Section 3.2, as evidenced by the accuracy trajectory graph on the left. On the other hand, five-layer models are capable of doing so.

Table 6: ICL on dual-pattern tasks, averaged over 10 repetitions, achieves notably better accuracy using learned than sinusoidal embeddings. Near-perfect accuracy is attained in the clean scenario by a 5-layer transformer with an embedding dimension (d_E) of 100 and learned positional embeddings. The block-noisy scenario is challenging; the same model attains the best performance.

		$d_E = 10$		$d_E = 100$	
	Pos. emb.	Clean	Block-noisy	Clean	Block-noisy
1-layer	Learned	(0.33, 0.33)	(0.15, 0.16)	(0.51, 0.49)	(0.49, 0.50)
	Sinusoidal	(0.12, 0.66)	(0.03, 0.03)	(0.51, 0.48)	(0.06, 0.10)
5-layer	Learned	(0.39, 0.39)	(0.23, 0.22)	(0.97, 0.98)	(0.87, 0.70)
	Sinusoidal	(0.32, 0.34)	(0.04, 0.04)	(0.83, 0.82)	(0.04, 0.07)

4 Scenarios where in-context learning fails

In this section, we consider two scenarios where in-context learning (ICL) fails, irrespective of architectures. In Section 4.1, we consider a logic reasoning task requiring identification and generalization of a repetition meta-pattern within sequences. In Section 4.2, we explore a word analogy task where relevant word pairs appear in unstructured training sentences but are limited to fixed positions. Section 4.3 concludes with a synthetic data experiment supporting the theory.

4.1 Failed scenario 1: Sentences with repeating patterns

In this meta-pattern recognition and generalization task, each training sequence follows a repeating pattern based on its starting tokens, and the ICL task sequence requires the model to identify this repetition and extend it to a new, unseen starting pattern. Specifically, our training data comprises sentences in the form of *abacdcefe*, where $a \neq b$, $c \neq d$, and $e \neq f$. Note that each sentence is structured into three blocks, each consisting of three tokens with the same pattern. For the ICL task, we consider predicting $\underline{f}$ from the prompt $\underline{abbcddef}$, where $\underline{a} \neq \underline{b}$, $\underline{c} \neq \underline{d}$, and $\underline{e} \neq \underline{f}$. Given the repeated pattern within each training sequence, a well-trained model might be expected to output $\underline{f}$ to continue the pattern established in the in-context examples: $\underline{abb}$ and $\underline{cdd}$. However, as seen in Table 7, all models fail to recognize and apply the pattern, resulting in incorrect predictions.

Formalization. We now formalize a generalization of this scenario. Let the vocabulary be $\mathcal{V} = \{1, 2, \cdots, |V|\}$, and define $S = \{(a, b) \mid a, b \in \mathcal{V}, a \neq b\}$. To ensure training sentences are distinct from the ICL prompts, we first partition S into S_1 and S_2, where $\{c[1] \mid c \in S_1\} = \{c[1] \mid c \in S_2\} = \mathcal{V}$. Here, $c[i]$ denotes the i-th element of c. Suppose we autoregressively train a sufficiently large position-aware model so that it is possible to achieve the minimum possible theoretical loss. The training sentences take the form $x_{11}x_{12}x_{11}x_{21}x_{22}x_{21} \cdots x_{N1}x_{N2}x_{N1}$, where $x_{i1} \neq x_{i2}$ and (x_{i1}, x_{i2}) is independently selected from S_1 for every $i \in [N]$. Theorem 5, whose proof is in Appendix H, states that ICL fails regardless of the number of in-context examples.

Theorem 5 (Failure of ICL: Different repeated patterns)**.** *Consider the generalized scenario in Section 4.1. For any $1 \leq \ell \leq N$, given an in-context prompt of the form $\underline{x_{11}x_{12}x_{12}x_{21}x_{22}x_{22}} \cdots \underline{x_{\ell 1}x_{\ell 2}}$ where $\underline{x_{i1}} \neq \underline{x_{i2}}$ and $(\underline{x_{i1}}, \underline{x_{i2}}) \in S_2$ for every $i \in [\ell]$, the model predicts $\underline{x_{\ell 1}}$ instead of $\underline{x_{\ell 2}}$.*

Results. Theorem 5 and Table 7 demonstrate that ICL achieves zero accuracy irrespective of the number of in-context examples ($\ell - 1$). This insight sheds light on the ICL capacity of autoregressive models. Simply put, if the pattern in the in-context examples differs significantly from any pattern in the training data, ICL may not occur. These results align with the findings of Raventós et al. [42] and Yadlowsky et al. [63] on the importance of data diversity for ICL.

4.2 Failed scenario 2: Sentences with co-occurring word pairs restricted to fixed locations

We revisit the word analogy task in Section 2. The training data now comprises sentences of the form of $a_i pqrsb_i$, where (a_i, b_i) represents a frequently co-occurring word pair and p, q, r, s represent other words. For the ICL task, we consider predicting b_{i_3} from the prompt $a_{i_1} b_{i_1} a_{i_2} b_{i_2} a_{i_3}$, where i_1, i_2, i_3 are distinct. As each training sentence always contains an (a_i, b_i) pair at a fixed location, we expect a well-trained model to output b_{i_3} to maintain the pattern in in-context examples: $a_{i_1} b_{i_1}$ and $a_{i_2} b_{i_2}$. Yet Table 7 shows none of the models can identify the patterns and predict the correct token.

Table 7: ICL in failed scenarios, averaged over 10 repetitions, achieves zero accuracy for any architecture and embedding dimension (d_E).

		Failed scenario 1		Failed scenario 2	
	Pos. emb.	$d_E = 10$	$d_E = 100$	$d_E = 10$	$d_E = 100$
1-layer	Learned	0.00	0.00	0.01	0.00
	Sinusoidal	0.01	0.00	0.00	0.00
5-layer	Learned	0.00	0.00	0.00	0.00
	Sinusoidal	0.00	0.00	0.00	0.00

Formalization. We now formalize a generalization of this scenario. Let the vocabulary be $\{(a_i, b_i)\}_{i \in [I]} \cup \mathcal{V}$, where $\mathcal{V} = \{1, 2, \cdots, |V|\}$ represent other words. As in Section 4.1, we autoregressively train a sufficiently large position-aware model that can achieve the minimum possible theoretical loss. The training sentences take the form $a_i v_1 v_2 \cdots v_{2k} b_i$, where i and $v_{1:2k}$ are independently chosen from $[I]$ and $\mathcal{V}$, respectively, uniformly at random. Theorem 6, whose proof is in Appendix I, states that ICL fails regardless of the number of in-context examples.

Theorem 6 (Failure of ICL: Different pattern structures)**.** *Consider the generalized scenario in Section 4.2. For any $1 \leq \ell \leq k+1$, given an in-context prompt of the form $a_{i_1} b_{i_1} a_{i_2} b_{i_2} \cdots a_{i_\ell}$ with distinct i_j's, the model never predicts b_{i_ℓ}: it predicts a uniform probability vector over $\mathcal{V}$ when $1 \leq \ell \leq k$, and b_{i_1} when $\ell = k+1$.*

Results. Theorem 6 highlights the finding that the success of ICL relies heavily on how the patterns appear in the training data. In this scenario, the (a_i, b_i) pairs consistently appear at the beginning and end of each training sentence, and we anticipate the model to recognize this relationship for ICL to occur. However, as shown in Theorem 6 and Table 7, this is not the case.

4.3 Experiment on a synthetic corpus

We conduct an experiment on a synthetic corpus featuring *(country)-(capital)* relationships. Each sentence falls into one of four categories: (1) exactly one country-capital pair, (2) exactly two country-capital pairs, (3) a single country without a pair, and (4) no country. In sentences with one country-capital pair, the capital appears in the first position, the country in the last, and each sentence contains six words (as in Section 4.2). The corpus generation process is detailed in Appendix K.

We train a five-layer two-head autoregressive transformer on this corpus, with an embedding dimension of 100. Similar to Section 2.4, we evaluate the ICL accuracies using prompts involving countries and their capitals. The results show zero ICL accuracy across varying in-context examples (one to five), supporting our theory.

5 Discussion

This paper examines how in-context learning (ICL) arises from pre-training on unstructured language data, with three key findings: (1) ICL for word analogy tasks can emerge from simple co-occurrence modeling, using models like continuous bag of words (CBOW) without positional encoding or attention; (2) positional information and structured nuisance tokens are essential for ICL in logic reasoning tasks that require recognizing rare patterns and generalizing to new tokens; and (3) the structure of training data significantly impacts ICL effectiveness.

Acknowledgements. This work was supported in part by the Office of Naval Research under grant number N00014-23-1-2590, the National Science Foundation under grant numbers 2231174 and 2310831, No. 2428059, and a Michigan Institute for Data Science Propelling Original Data Science (PODS) grant.

References

[1] M. Abbas, Y. Zhou, P. Ram, N. Baracaldo, H. Samulowitz, T. Salonidis, and T. Chen. Enhancing in-context learning via linear probe calibration. In *Artificial Intelligence and Statistics*, 2024.

[2] J. Abernethy, A. Agarwal, T. V. Marinov, and M. K. Warmuth. A mechanism for sample-efficient in-context learning for sparse retrieval tasks. In *Algorithmic Learning Theory*, 2024.

[3] K. Ahn, X. Cheng, H. Daneshmand, and S. Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Neural Information Processing Systems*, 2024.

[4] K. Ahuja and D. Lopez-Paz. A closer look at in-context learning under distribution shifts. In *Workshop on Efficient Systems for Foundation Models at ICML*, 2023.

[5] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou. What learning algorithm is in-context learning? Investigations with linear models. In *International Conference on Learning Representations*, 2022.

[6] E. Akyürek, B. Wang, Y. Kim, and J. Andreas. In-context language learning: Architectures and algorithms. *arXiv preprint arXiv:2401.12973*, 2024.

[7] Y. Bai, F. Chen, H. Wang, C. Xiong, and S. Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Neural Information Processing Systems*, 2023.

[8] S. Bhattamishra, A. Patel, P. Blunsom, and V. Kanade. Understanding in-context learning in transformers and LLMs by learning to learn discrete functions. In *International Conference on Learning Representations*, 2023.

[9] A. Bietti, V. Cabannes, D. Bouchacourt, H. Jegou, and L. Bottou. Birth of a transformer: A memory viewpoint. In *Neural Information Processing Systems*, 2023.

[10] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Neural Information Processing Systems*, 2020.

[11] S. C. Chan, A. Santoro, A. K. Lampinen, J. X. Wang, A. K. Singh, P. H. Richemond, J. McClelland, and F. Hill. Data distributional properties drive emergent in-context learning in transformers. In *Neural Information Processing Systems*, 2022.

[12] S. Chen, H. Sheen, T. Wang, and Z. Yang. Training dynamics of multi-head softmax attention for in-context learning: Emergence, convergence, and optimality. *arXiv preprint arXiv:2402.19442*, 2024.

[13] Y. Chen, C. Zhao, Z. Yu, K. McKeown, and H. He. Parallel structures in pre-training data yield in-context learning. *arXiv preprint arXiv:2402.12530*, 2024.

[14] T.-R. Chiang and D. Yogatama. Understanding in-context learning with a pelican soup framework. *arXiv preprint arXiv:2402.10424*, 2024.

[15] L. Collins, A. Parulekar, A. Mokhtari, S. Sanghavi, and S. Shakkottai. In-context learning with transformers: Softmax attention adapts to function Lipschitzness. *arXiv preprint arXiv:2402.11639*, 2024.

[16] Y. Cui, J. Ren, P. He, J. Tang, and Y. Xing. Superiority of multi-head attention in in-context linear regression. *arXiv preprint arXiv:2401.17426*, 2024.

[17] D. Dai, Y. Sun, L. Dong, Y. Hao, Z. Sui, and F. Wei. Why can GPT learn in-context? Language models secretly perform gradient descent as meta optimizers. In *Association for Computational Linguistics*, 2023.

[18] S. Dalal and V. Misra. The matrix: A Bayesian learning model for LLMs. *arXiv preprint arXiv:2402.03175*, 2024.

[19] N. Ding, T. Levinboim, J. Wu, S. Goodman, and R. Soricut. CausalLM is not optimal for in-context learning. In *International Conference on Learning Representations*, 2024.

[20] D. Fu, T.-Q. Chen, R. Jia, and V. Sharan. Transformers learn higher-order optimization methods for in-context learning: A study with linear models. In *Workshop on Mathematics of Modern Machine Learning at NeurIPS*, 2023.

[21] S. Garg, D. Tsipras, P. S. Liang, and G. Valiant. What can transformers learn in-context? A case study of simple function classes. In *Neural Information Processing Systems*, 2022.

[22] T. Guo, W. Hu, S. Mei, H. Wang, C. Xiong, S. Savarese, and Y. Bai. How do transformers learn in-context beyond simple functions? A case study on learning with representations. In *International Conference on Learning Representations*, 2023.

[23] M. Hahn and N. Goyal. A theory of emergent in-context learning as implicit structure induction. *arXiv preprint arXiv:2303.07971*, 2023.

[24] C. Han, Z. Wang, H. Zhao, and H. Ji. Explaining emergent in-context learning as kernel regression. *arXiv preprint arXiv:2305.12766*, 2023.

[25] X. Han, D. Simig, T. Mihaylov, Y. Tsvetkov, A. Celikyilmaz, and T. Wang. Understanding in-context learning via supportive pretraining data. In *Association for Computational Linguistics*, 2023.

[26] A. Haviv, O. Ram, O. Press, P. Izsak, and O. Levy. Transformer language models without positional encodings still learn positional information. In *Empirical Methods in Natural Language Processing*, 2022.

[27] Y. Huang, Y. Cheng, and Y. Liang. In-context convergence of transformers. In *Workshop on Mathematics of Modern Machine Learning at NeurIPS*, 2023.

[28] H. J. Jeon, J. D. Lee, Q. Lei, and B. Van Roy. An information-theoretic analysis of in-context learning. *arXiv preprint arXiv:2401.15530*, 2024.

[29] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[30] J. Kossen, Y. Gal, and T. Rainforth. In-context learning learns label relationships but is not conventional learning. In *International Conference on Learning Representations*, 2024.

[31] S. Li, Z. Song, Y. Xia, T. Yu, and T. Zhou. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*, 2023.

[32] X. Li and X. Qiu. Finding support examples for in-context learning. In *Empirical Methods in Natural Language Processing*, 2023.

[33] Y. Li, M. E. Ildiz, D. Papailiopoulos, and S. Oymak. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, 2023.

[34] Z. Lin and K. Lee. Dual operating modes of in-context learning. *arXiv preprint arXiv:2402.18819*, 2024.

[35] A. V. Mahankali, T. Hashimoto, and T. Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. In *International Conference on Learning Representations*, 2023.

[36] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[37] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Empirical Methods in Natural Language Processing*, 2022.

[38] C. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma, T. Henighan, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, S. Johnston, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022.

[39] M. Panwar, K. Ahuja, and N. Goyal. In-context learning through the Bayesian prism. In *International Conference on Learning Representations*, 2023.

[40] K. Peng, L. Ding, Y. Yuan, X. Liu, M. Zhang, Y. Ouyang, and D. Tao. Revisiting demonstration selection strategies in in-context learning. *arXiv preprint arXiv:2401.12087*, 2024.

[41] C. Qin, A. Zhang, A. Dagar, and W. Ye. In-context learning with iterative demonstration selection. *arXiv preprint arXiv:2310.09881*, 2023.

[42] A. Raventós, M. Paul, F. Chen, and S. Ganguli. Pretraining task diversity and the emergence of non-Bayesian in-context learning for regression. In *Neural Information Processing Systems*, 2023.

[43] J. Ren, Q. Guo, H. Yan, D. Liu, X. Qiu, and D. Lin. Identifying semantic induction heads to understand in-context learning. *arXiv preprint arXiv:2402.13055*, 2024.

[44] R. Ren and Y. Liu. In-context learning with transformer is really equivalent to a contrastive learning pattern. *arXiv preprint arXiv:2310.13220*, 2023.

[45] M. E. Sander, R. Giryes, T. Suzuki, M. Blondel, and G. Peyré. How do transformers perform in-context autoregressive learning? *arXiv preprint arXiv:2402.05787*, 2024.

[46] L. Shen, A. Mishra, and D. Khashabi. Do pretrained transformers really learn in-context by gradient descent? *arXiv preprint arXiv:2310.08540*, 2023.

[47] A. Singh, S. Chan, T. Moskovitz, E. Grant, A. Saxe, and F. Hill. The transient nature of emergent in-context learning in transformers. In *Neural Information Processing Systems*, 2023.

[48] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

[49] S. Swaminathan, A. Dedieu, R. Vasudeva Raju, M. Shanahan, M. Lazaro-Gredilla, and D. George. Schema-learning and rebinding as mechanisms of in-context learning and emergence. In *Neural Information Processing Systems*, 2023.

[50] E. Todd, M. Li, A. Sharma, A. Mueller, B. C. Wallace, and D. Bau. Function vectors in large language models. In *International Conference on Learning Representations*, 2024.

[51] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[52] M.-H. Van, X. Wu, et al. In-context learning demonstration selection via influence analysis. *arXiv preprint arXiv:2402.11750*, 2024.

[53] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.

[54] M. Vladymyrov, J. von Oswald, M. Sandler, and R. Ge. Linear transformers are versatile in-context learners. *arXiv preprint arXiv:2402.14180*, 2024.

[55] J. Von Oswald, E. Niklasson, E. Randazzo, J. Sacramento, A. Mordvintsev, A. Zhmoginov, and M. Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, 2023.

[56] X. Wang, W. Zhu, M. Saxon, M. Steyvers, and W. Y. Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Neural Information Processing Systems*, 2023.

[57] K. C. Wibisono and Y. Wang. On the role of unstructured training data in transformers' in-context learning capabilities. In *Workshop on Mathematics of Modern Machine Learning at NeurIPS*, 2023.

[58] N. Wies, Y. Levine, and A. Shashua. The learnability of in-context learning. In *Neural Information Processing Systems*, 2023.

[59] J. Wu, D. Zou, Z. Chen, V. Braverman, Q. Gu, and P. Bartlett. How many pretraining tasks are needed for in-context learning of linear regression? In *International Conference on Learning Representations*, 2023.

[60] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma. An explanation of in-context learning as implicit Bayesian inference. In *International Conference on Learning Representations*, 2021.

[61] Y. Xing, X. Lin, N. Suh, Q. Song, and G. Cheng. Benefits of transformer: In-context learning in linear regression tasks with unstructured data. *arXiv preprint arXiv:2402.00743*, 2024.

[62] Z. Xu, Z. Shi, and Y. Liang. Do large language models have compositional ability? An investigation into limitations and scalability. In *Workshop on Mathematical and Empirical Understanding of Foundation Models at ICLR*, 2024.

[63] S. Yadlowsky, L. Doshi, and N. Tripuraneni. Pretraining data mixtures enable narrow model selection capabilities in transformer models. *arXiv preprint arXiv:2311.00871*, 2023.

[64] J. Yan, J. Xu, C. Song, C. Wu, Y. Li, and Y. Zhang. Understanding in-context learning from repetitions. In *International Conference on Learning Representations*, 2023.

[65] Z. Yu and S. Ananiadou. How do large language models learn in-context? Query and key matrices of in-context heads are two towers for metric learning. *arXiv preprint arXiv:2402.02872*, 2024.

[66] R. Zhang, S. Frei, and P. L. Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 2024.

[67] Y. Zhang, F. Zhang, Z. Yang, and Z. Wang. What and how does in-context learning learn? Bayesian model averaging, parameterization, and generalization. *arXiv preprint arXiv:2305.19420*, 2023.

[68] Y. Zhao, Y. Sakai, and N. Inoue. NoisyICL: A little noise in model parameters calibrates in-context learning. *arXiv preprint arXiv:2402.05515*, 2024.

Supplementary Material

A Related work

Large language models (LLMs), such as transformers, are widely recognized for their outstanding performance in in-context learning (ICL) [10]. ICL refers to the capability of LLMs to discern specific tasks and generate predictions based on prompt exemplars without needing any parameter updates. A multitude of studies have been dedicated to exploring this intriguing phenomenon from various theoretical and empirical perspectives. In this section, we provide a brief summary of some of these studies.

Some studies adopted a Bayesian approach to studying ICL. Xie et al. [60] posited that ICL can be viewed as implicit Bayesian inference. They demonstrated that LLMs can infer a latent document-level concept for next-token prediction during pre-training and a shared latent concept across input-output pairs in an ICL prompt, under the assumption that documents are generated from hidden Markov models (HMMs). Wang et al. [56] and Zhang et al. [67] expanded on this idea by exploring more realistic latent variable models beyond HMMs. Wang et al. [56] argued that large language models function as latent variable models, with latent variables containing task-related information being implicitly inferred. Zhang et al. [67] showed that without updating the neural network parameters, ICL can be interpreted as Bayesian model averaging parameterized by the attention mechanism. Panwar et al. [39] provided empirical evidence that transformers behave like Bayesian predictors when performing ICL with linear and non-linear function classes. Dalal and Misra [18] proposed a Bayesian learning framework to understand ICL through the lens of text generation models represented by multinomial transition probability matrices. Chiang and Yogatama [14] proposed the pelican soup framework to explain ICL without relying on latent variable models. This framework incorporates concepts such as a common sense knowledge base, natural language classification, and meaning association, enabling the establishment of a loss bound for ICL that depends on the number of in-context examples.

Garg et al. [21] formulated ICL as learning a specific function class $\mathcal{F}$ from prompts of the form $(x_1, f(x_1), \ldots, x_n, f(x_n), x_{n+1})$ and their corresponding responses $f(x_{n+1})$. Here, $f \in \mathcal{F}$, where $\mathcal{F}$ is a function class. In this context, ICL refers to the capability of a transformer to output a number close to $g(y_{n+1})$ given a prompt of the form $(y_1, g(y_1), \ldots, y_n, g(x_n), y_{n+1})$, where $g \in \mathcal{F}$. Many studies adopted this regression formulation of ICL, with some linking ICL to gradient descent. Akyürek et al. [5], Von Oswald et al. [55], and Dai et al. [17] proved that transformers are capable of implementing gradient descent, which results in their ICL ability. Bai et al. [7] established generalization bounds for ICL and proved that transformers can perform algorithm selection like statisticians. Zhang et al. [66] showed that the gradient flow dynamics of transformers converge to a global minimum that enables ICL. Huang et al. [27] investigated the learning dynamics of single-layer softmax transformers trained via gradient descent to perform ICL on linear functions. Ahn et al. [3] explored the optimization landscape of transformers and proved that the optimal parameters coincide with an iteration of preconditioned gradient descent.

In a related exploration, Li et al. [31] showed that softmax regression models learned through gradient descent are similar to transformers. Ren and Liu [44] related ICL with softmax transformers to contrastive learning, where the inference process of ICL can be viewed as a form of gradient descent. Mahankali et al. [35] proved that minimizing the pre-training loss is equivalent to a step of gradient descent in single-layer linear transformers. Vladymyrov et al. [54] established that linear transformers execute a variant of preconditioned gradient descent by maintaining implicit linear models. On the other hand, some studies argued that the ICL ability of transformers cannot be attributed to gradient descent. Fu et al. [20] showed that ICL for linear regression tasks arises from higher-order optimization techniques like iterative Newton's method rather than gradient descent. Wibisono and Wang [57] demonstrated that transformers can perform ICL on unstructured data whose prompt exemplars lack explicit pairings, with softmax attention playing an important role especially when using a single attention layer. Shen et al. [46] provided empirical evidence that the equivalence between gradient descent and ICL might not be applicable in real-world scenarios. *In contrast to these studies, our work provides a connection between ICL and classical language models like continuous bag of words (CBOW). Specifically, we show that ICL for word analogy tasks with semantically related word pairs can arise by modeling co-occurrence patterns via CBOW.*

Numerous studies focused on the pre-training aspects (e.g., data distribution and task diversity) of ICL. Min et al. [37] showed that the input-label mapping in the in-context examples does not significantly affect ICL performance. Chan et al. [11] demonstrated that the ICL capabilities of transformers depend on the training data distributions and model features. Kossen et al. [30] established that ICL considers in-context label information and is capable of learning entirely new tasks in-context. Li and Qiu [32] introduced an iterative algorithm designed to enhance ICL performance by selecting a small set of informative examples that effectively characterize the ICL task. Qin et al. [41] proposed a method based on zero-shot chain-of-thought reasoning for selecting ICL examples, emphasizing the importance of choosing diverse examples that are strongly correlated with the test sample. Han et al. [25] studied ICL by identifying a small subset of the pre-training data that support ICL via gradient-based methods. They discovered that this supportive pre-training data typically consist of more uncommon tokens and challenging examples, characterized by a small information gain from long-range context. Peng et al. [40] proposed a selection method for ICL demonstrations that are both data-dependent and model-dependent. Van et al. [52] introduced a demonstration selection method that enhances ICL performance by analyzing the influences of training samples using influence functions.

In a similar vein, Wu et al. [59] demonstrated that pre-training single-layer linear attention models for ICL on linear regression with a Gaussian prior can be effectively accomplished with a minimal number of independent tasks, regardless of task dimension. Raventós et al. [42] emphasized a task diversity threshold that differentiates the conditions under which transformers can successfully address unseen tasks. Yadlowsky et al. [63] attributed the impressive ICL capabilities of transformers to the diversity and range of data mixtures in their pre-training, rather than their inductive biases for generalizing to new tasks. Ding et al. [19] compared the ICL performance of transformers trained with prefixLM (where in-context samples can attend to all tokens) versus causalLM (where in-context samples cannot attend to subsequent tokens), finding that the latter resulted in poorer ICL performance. Chen et al. [13] discovered that the ICL capabilities of language models rely on the presence of pairs of phrases with similar structures within the same sentence. Zhao et al. [68] proposed a calibration scheme that modifies model parameters by adding random noises, resulting in fairer and more confident predictions. Abbas et al. [1] demonstrated that the ICL predictions from transformer-based models often exhibit low confidence, as indicated by high Shannon entropy. To address this issue, they introduced a straightforward method that linearly calibrates output probabilities, independent of the model's weights or architecture. *Similar to these works, our work highlights the importance of co-occurrence, positional information, and training data structure for ICL to arise.*

Other studies analyzed ICL from a learning theory perspective. Hahn and Goyal [23] proposed an information-theoretic bound that explains how ICL emerges from next-token prediction. Wies et al. [58] derived a PAC-type framework for ICL and finite-sample complexity results. Jeon et al. [28] introduced a novel information-theoretic view of meta-learning (including ICL), allowing for the decomposition of errors into three components. They proved that in ICL, the errors decrease as the number of examples or sequence length increase. Other studies focus on the mechanistic interpretability component of ICL. Olsson et al. [38] argued that transformers can develop induction heads that are able to complete token sequences such as $[A][B] \cdots [A] \rightarrow [B]$, leading to impressive ICL performance. Bietti et al. [9] examined a setup where tokens are generated from either global or context-specific bigram distributions to distinguish between global and in-context learning. They found that global learning occurs rapidly, while in-context learning is achieved gradually through the development of an induction head. Ren et al. [43] identified semantic induction heads that increase the output logits of tail tokens when attending to head tokens, providing evidence that these heads could play a vital role in the emergence of ICL. Yu and Ananiadou [65] showed that the ICL ability of transformers arises from the utilization of in-context heads, where each query and key matrix collaborate to learn the similarity between the input text and each demonstration example.

A number of works delved into specific data generating processes to provide insight into the emergence of ICL. Bhattamishra et al. [8] examined the ICL ability of transformers by focusing on discrete functions. Specifically, they showed that transformers perform well on simpler tasks, struggle with more complex tasks, and can learn more efficiently when provided with examples that uniquely identify a task. Guo et al. [22] investigated ICL in scenarios where each label is influenced by the input through a potentially complex yet constant representation function, coupled with a unique linear function for each instance. Akyürek et al. [6] studied ICL of regular languages produced by random finite automata. They compared numerous neural sequence models and demonstrated that

transformers significantly outperform RNN-based models because of their ability to develop *n-gram heads*, which are a generalization of *induction heads*. Sander et al. [45] analyzed simple first-order autoregressive processes to gain insight into how transformers perform ICL to predict the next tokens. *On the other hand, our work focuses on data generating processes containing semantically related word pairs and repeated token patterns to better understand several components that are crucial for ICL to occur from training on unstructured data.*

Some studies explored how different components of transformers affect their ICL abilities. Ahuja and Lopez-Paz [4] compared the ICL performance of transformers and MLP-based architectures under distribution shifts. Their findings demonstrate that while both methods perform well in in-distribution ICL, transformers exhibit superior ICL performance when faced with mild distribution shifts. Collins et al. [15] showed that softmax attention outperforms linear attention in ICL due to its ability to calibrate its attention window to the Lipschitzness of the pre-training tasks. Xing et al. [61] focused on linear regression tasks to identify transformer components that enable ICL. They found that positional encoding is crucial, along with the use of multiple heads, multiple layers, and larger input dimensions. Cui et al. [16] proved that multi-head attention outperforms single-head attention in various practical scenarios, including those with noisy labels and correlated features. Chen et al. [12] investigated the ICL dynamics of a multi-head softmax attention model applied to multi-task linear regression. They proved the convergence of the gradient flow and observed the emergence of a *task allocation* phenomenon, where each attention head specializes in a specific task.

Finally, several studies proposed various hypotheses on the emergence of ICL and provided theoretical justifications. Swaminathan et al. [49] introduced clone-structured causal graphs (CSCGs) to explain how ICL can generalize to unseen sentences via a mechanism called rebinding. Li et al. [33] viewed ICL as an algorithm learning problem where a transformer implicitly constructs a hypothesis function at inference time. Han et al. [24] argued that the ability of transformers to execute ICL is attributable to their capacity to simulate kernel regression. Singh et al. [47] explored the interaction between ICL and in-weights learning (IWL) using synthetic data designed to support both processes. They observed that ICL initially emerges, followed by a transient phase where it disappears and gives rise to IWL. Yan et al. [64] studied ICL from the perspective that token co-occurrences play a crucial role in guiding the learning of surface patterns that facilitates ICL. Abernethy et al. [2] showed that transformers can execute ICL by dividing a prompt into examples and labels, then employing sparse linear regression to deduce input-output relationships and generate predictions. Lin and Lee [34] developed a probabilistic model that can simultaneously explain both task learning and task retrieval aspects of ICL. Here, task learning refers to the ability of language models to identify a task from in-context examples, while task retrieval pertains to their ability to locate the relevant task within the pre-training data.

B Proof of Theorem 1

Proof. Let $|V| = 2K + L$ denote the vocabulary size. Consider a sentence X represented by its one-hot encoding (i.e., $X \in \{0,1\}^{|V|\times S}$). For every position $i \in [S]$, the loss for predicting the word in the i-th position given all the other words is given by $||AX(\mathbb{1}_S - e_i) - Xe_i||_2^2$, where $A = \frac{U^\top V}{S-1} \in \mathbb{R}^{|V|\times|V|}$ and $e_i \in \mathbb{R}^S$ is a zero vector with 1 on its i-th entry. Here, U (V) is a matrix consisting of the center (context) embeddings of all tokens, and A is a matrix summarizing the similarity between each pair of words (one as a center word and the other as a context word). Our objective is to find A that minimizes the sum of losses for each position in each sentence. Lemma 1 gives a closed-form expression of the minimizer.

Lemma 1. *The minimizer of the overall loss is given by $A = B\left((S-2)B + C\right)^{-1}$. Here, B is a matrix whose (i,j)-th entry is $p(i,j)$, the probability that for a given (center, context) pair, the center is $i \in |V|$ and the context is $j \in |V|$. Moreover, C is a diagonal matrix whose i-th diagonal entry is $p(i) = \sum_{j\in|V|} p(i,j)$.*

Proof. Let $\mathcal{L}(X) = \sum_{i=1}^{S} ||AX(\mathbb{1}_S - e_i) - Xe_i||_2^2$ denote the sum of the losses corresponding to all tokens in sentence X. By direct calculation,

$$\frac{\partial \mathcal{L}(X)}{\partial A} = 2AX\left(\sum_{i=1}^{S}(\mathbb{1}_S - e_i)(\mathbb{1}_S - e_i)^\top\right)X^\top - 2X\left(\sum_{i=1}^{S} e_i(\mathbb{1}_S - e_i)^\top\right)X^\top$$

.

Note that $\sum_{i=1}^{S}(\mathbb{1}_S - e_i)(\mathbb{1}_S - e_i)^\top = (S-2)\mathbb{1}_{S\times S} + \mathbb{I}_{S\times S}$ and $\sum_{i=1}^{S} e_i(\mathbb{1}_S - e_i)^\top = \mathbb{1}_{S\times S} - \mathbb{I}_{S\times S}$. Now, let our sentences be $X_1, X_2, \cdots, X_N$. The minimizer of the overall loss thus satisfies

$$A\,\frac{1}{N}\sum_{k=1}^{N} X_k\left((S-2)\mathbb{1}_{S\times S} + \mathbb{I}_{S\times S}\right)X_k^\top = \frac{1}{N}\sum_{k=1}^{N} X_k\left(\mathbb{1}_{S\times S} - \mathbb{I}_{S\times S}\right)X_k^\top. \tag{1}$$

We denote the number of (center, context) pairs across all sentences in which the center is i and the context is j by $\#(i,j)$. Moreover, we define $\#(i) = \sum_{j\in|V|}\#(i,j)$. It is easy to see that Equation (1) can be rewritten as

$$A\left((S-2)\tilde{B} + \tilde{C}\right) = \tilde{B},$$

where $\tilde{B}$ is a matrix such that its (i,j)-th entry is $\frac{\#(i,j)}{N}$ and $\tilde{C}$ is a diagonal matrix such that its i-th diagonal element is $\frac{\#(i)}{N}$. As $N \to \infty$, an application of the law of large numbers yields $\frac{\#(i,j)}{N} \to S(S-1)p(i,j)$ almost surely and $\frac{\#(i)}{N} \to S(S-1)p(i)$ almost surely, where $p(i,j)$ is the probability that for a given (center, context) pair, the center is i and the context is j, and $p(i) = \sum_{j\in|V|} p(i,j)$.

Thus, as $N \to \infty$, we have

$$A = B\left((S-2)B + C\right)^{-1},$$

where B and C are defined in the statement of Lemma 1. □

We now define

- $p_1 = p(c_i, c_j) = p(d_i, d_j) = p(c_i, d_j) = p(d_i, c_j)$ for any $i \neq j$;
- $p_2 = p(r_i, r_j)$ for any $i \neq j$;
- $p_3 = p(c_i, d_i) = p(d_i, c_i)$ for any i;
- $p_4 = p(c_i, r_j) = p(d_i, r_j) = p(r_j, c_i) = p(r_j, d_i)$ for any i, j,

where the equalities in the probabilities are a consequence of the data distribution.

For ease of presentation, we denote a square matrix with α on the diagonal and β off the diagonal as $X_{\alpha,\beta}$, and a matrix with all entries γ as Y_γ. We then have

$$B = \begin{bmatrix} X_{0,p_1} & X_{p_3,p_1} & Y_{p_4} \\ X_{p_3,p_1} & X_{0,p_1} & Y_{p_4} \\ Y_{p_4} & Y_{p_4} & X_{0,p_2} \end{bmatrix}.$$

Now, define $a = (S-2)p_1$, $b = (S-2)p_2$, $c = (S-2)p_3$, $d = (S-2)p_4$, $e = 2(K-1)p_1 + p_3 + Lp_4$, and $f = (L-1)p_2 + 2Kp_4$. It is easy to see that

$$(S-2)B + C = \begin{bmatrix} X_{e,a} & X_{c,a} & Y_d \\ X_{c,a} & X_{e,a} & Y_d \\ Y_d & Y_d & X_{f,b} \end{bmatrix}.$$

Moreover, its inverse can be written as

$$((S-2)B + C)^{-1} = \begin{bmatrix} X_{q_5,q_1} & X_{q_3,q_1} & Y_{q_4} \\ X_{q_3,q_1} & X_{q_5,q_1} & Y_{q_4} \\ Y_{q_4} & Y_{q_4} & X_{q_6,q_2} \end{bmatrix},$$

where

$\Delta = 2a(K-1)(b(L-1)+f) + b(L-1)(c+e) + cf - 2d^2KL + ef,$

$q_1 = -\left(\frac{-abL+ab-af+d^2L}{(2a-c-e)\Delta}\right),$

$q_2 = \frac{2ab(K-1)+b(c+e)-2d^2K}{(b-f)\Delta},$

$$q_3 = -\left(\frac{\begin{array}{r}-2a^2b(K-1)(L-1) - 2a^2f(K-1) + 2abc(K-2)(L-1) + 2acf(K-2) \\ + 2(a-c)d^2KL + bc(c+e)(L-1) + cf(c+e) + d^2L(c-e)\end{array}}{(c-e)(2a-c-e)\Delta}\right),$$

$q_4 = -\left(\frac{d}{\Delta}\right),$

$$q_5 = -\left(\frac{\begin{array}{r}-2a^2b(K-1)(L-1) - 2a^2f(K-1) + 2abe(K-2)(L-1) + 2aef(K-2) \\ + 2(a-e)d^2KL + be(c+e)(L-1) + ef(c+e) + d^2L(e-c)\end{array}}{(e-c)(2a-c-e)\Delta}\right),$$

and $q_6 = -\left(\frac{2a(K-1)(b(L-2)+f)+b(L-2)(c+e)+cf-2d^2KL+2d^2K+ef}{(b-f)\Delta}\right).$

By computing $A = B((S-2)B + C)^{-1}$, given the following center words, the similarities between them and all possible context words are as follows:

- Center word = c_i for any i
  - $c_i : 2(K-1)p_1q_1 + p_3q_3 + Lp_4q_4$;
  - $c_j : 2(K-2)p_1q_1 + p_1q_5 + p_3q_1 + p_1q_3 + Lp_4q_4$ $(j \neq i)$;
  - $d_i : 2(K-1)p_1q_1 + p_3q_5 + Lp_4q_4$;
  - $d_j : 2(K-2)p_1q_1 + p_1q_3 + p_3q_1 + p_1q_5 + Lp_4q_4$ $(j \neq i)$;
  - $r_j : 2(K-1)p_1q_4 + p_3q_4 + p_4q_6 + (L-1)p_4q_2$ (for any j).
- Center word = d_i for any i
  - $d_i : 2(K-1)p_1q_1 + p_3q_3 + Lp_4q_4$;
  - $d_j : 2(K-2)p_1q_1 + p_1q_5 + p_3q_1 + p_1q_3 + Lp_4q_4$ $(j \neq i)$;
  - $c_i : 2(K-1)p_1q_1 + p_3q_5 + Lp_4q_4$;

- $c_j : 2(K-2)p_1q_1 + p_1q_3 + p_3q_1 + p_1q_5 + Lp_4q_4$ ($j \neq i$);
- $r_j : 2(K-1)p_1q_4 + p_3q_4 + p_4q_6 + (L-1)p_4q_2$ (for any j).

- Center word = r_i
  - $c_j : 2(K-1)p_4q_1 + p_4q_5 + p_4q_3 + (L-1)p_2q_4$ (for any j);
  - $d_j : 2(K-1)p_4q_1 + p_4q_5 + p_4q_3 + (L-1)p_2q_4$ (for any j);
  - $r_i : 2Kp_4q_4 + (L-1)p_2q_2$;
  - $r_j : 2Kp_4q_4 + (L-2)p_2q_2 + p_2q_6$ ($j \neq i$).

Recall that the ICL problem of interest is the following: given context words $c_{i_1}d_{i_1}\cdots c_{i_\ell}d_{i_\ell}c_{i_{\ell+1}}$, we aim to predict $d_{i_{\ell+1}}$. Without loss of generality, we can rewrite the problem to predict $d_{\ell+1}$ given context words $c_1d_1\cdots c_\ell d_\ell c_{\ell+1}$. We now compute the total similarity for each possible center word, where $\epsilon^\top\delta$ indicates the similarity between the word ϵ in the center and the word δ in the context.

- c_1 (or any of $c_2, \cdots, c_\ell$) : $c_1^\top c_1 + \ell c_1^\top c_2 + c_1^\top d_1 + (\ell-1)c_1^\top d_2$;
- d_1 (or any of $d_2, \cdots, d_\ell$) : $c_1^\top d_1 + \ell c_1^\top d_2 + c_1^\top c_1 + (\ell-1)c_1^\top c_2$;
- r_1 (or any other r_k's) : $(\ell+1)r_1^\top c_1 + \ell r_1^\top d_1 = (2\ell+1)r_1^\top c_1$;
- $c_{\ell+1}$: $\ell c_1^\top c_2 + \ell c_1^\top d_2 + c_1^\top c_1$;
- $d_{\ell+1}$: $\ell c_1^\top d_2 + \ell c_1^\top c_2 + c_1^\top d_1$;
- $c_{\ell+2}$ (or any c_k's not in the context prompt) : $(\ell+1)c_1^\top c_2 + \ell c_1^\top d_2$;
- $d_{\ell+2}$ (or any d_k's not in the context prompt) : $(\ell+1)c_1^\top d_2 + \ell c_1^\top c_2$.

Note that correctly predicting $d_{\ell+1}$ is equivalent to the following conditions being simultaneously satisfied:

- $c_1^\top d_1 > c_1^\top c_1$, equivalent to $p_3q_5 > p_3q_3$;
- $c_1^\top d_2 > c_1^\top c_1$ and $c_1^\top c_2 > c_1^\top c_1$, equivalent to $p_1q_3 + p_3q_1 + p_1q_5 > 2p_1q_1 + p_3q_3$;
- $c_1^\top d_1 > c_1^\top c_2$ and $c_1^\top d_1 > c_1^\top d_2$, equivalent to $2p_1q_1 + p_3q_5 \geq p_1q_5 + p_1q_3 + p_3q_1$;
- $2\ell c_1^\top c_2 + c_1^\top d_1 > (2\ell+1)r_1^\top c_1$, equivalent to $2\ell(2(K-2)p_1q_1 + p_1q_5 + p_3q_1 + p_1q_3 + Lp_4q_4) + 2(K-1)p_1q_1 + p_3q_5 + Lp_4q_4 > (2\ell+1)(2(K-1)p_4q_1 + p_4q_5 + p_4q_3 + (L-1)p_2q_4)$;

In our data generating process, it is easy to see that $p_1 = 0$, $p_2 = \frac{(S-2)(S-3)}{L(L-1)}$, $p_3 = \frac{1}{K}$, and $p_4 = \frac{S-2}{KL}$, where each p_i is multiplied by a constant $S(S-1) > 0$ (without loss of generalization) to make calculations easier. From here, we have $a = 0$, $b = \frac{(S-2)^2(S-3)}{L(L-1)}$, $c = \frac{S-2}{K}$, $d = \frac{(S-2)^2}{KL}$, $e = \frac{S-1}{K}$, and $f = \frac{(S-1)(S-2)}{L}$. Substituting to the above, we have

- $q_1 = \frac{(S-2)^4}{\Delta KL(2S-3)}$;
- $q_3 = \frac{-K(S-2)^2(S-1)^2-(S-2)^4}{\Delta KL(2S-3)}$;
- $q_4 = \frac{-(2S-3)(S-2)^2}{\Delta KL(2S-3)}$;
- $q_5 = \frac{K(S-2)(S-1)^3+(S-2)^4}{\Delta KL(2S-3)}$,

where $\Delta = \frac{(S-1)^2(S-2)}{KL} > 0$.

We now check when these conditions are simultaneously satisfied. The first condition is equivalent to $p_3 > 0$ and $K > \frac{2(S-2)^3}{(S-1)^2(2S-3)}$, which always hold. The second condition reduces to $p_3 > 0$ and $2(S-2)^4 + K(S-2)^2(S-1)^2 > 0$, which is also true. The third condition can be written as $p_3 > 0$ and $K(S-2)(S-1)^3 > 0$, which always hold. The last condition becomes

$$(2\ell+1)((K+L)(S-2)^2(S-1) + K(S-2)(S-1)^2 - 2(S-2)^4) < KL(S-1)^3,$$

which is equivalent to

$$2\ell + 1 < \frac{KL(S-1)^3}{(K+L)(S-2)^2(S-1) + K(S-2)(S-1)^2 - 2(S-2)^4},$$

completing the proof.

Note that this condition ensures that the model predicts $d_{\ell+1}$ instead of one of the r_i's. □

C Comparison of ICL performance using squared and cross-entropy loss across different numbers of examples

Table 8: ICL performance in the *clean* scenario, evaluated with both squared and cross-entropy loss functions across different numbers of examples (0 to 8) with $d_E = 100$, averaged over 10 repetitions.

	Squared					Cross-entropy				
(p_0, p_1, p_2)	0	2	4	6	8	0	2	4	6	8
$(0, 1, 0)$	1	1	0	0	0	0.87	0	0	0	0
$(0, 0, 1)$	1	1	1	0	0	1	0	0	0	0
$(1/2, 1/2, 0)$	1	1	1	1	1	1	1	0.34	0	0
$(1/2, 0, 1/2)$	1	1	1	1	1	1	1	1	1	1
$(0, 1/2, 1/2)$	1	1	1	1	1	1	1	1	0	0
$(1/3, 1/3, 1/3)$	1	1	1	1	1	1	1	1	1	0

Table 9: ICL performance in the *corrupted* scenario, evaluated with both squared and cross-entropy loss functions across different numbers of examples (0 to 8) with $d_E = 100$, averaged over 10 repetitions.

	Squared					Cross-entropy				
(p_0, p_1, p_2)	0	2	4	6	8	0	2	4	6	8
$(0, 1, 0)$	1	0	0	0	0	0	0	0	0	0
$(0, 0, 1)$	1	0.97	0	0	0	1	0	0	0	0
$(1/2, 1/2, 0)$	1	1	1	0.53	0	1	0	0	0	0
$(1/2, 0, 1/2)$	1	1	1	1	1	1	1	1	1	1
$(0, 1/2, 1/2)$	1	1	0.76	0	0	1	1	0	0	0
$(1/3, 1/3, 1/3)$	1	1	1	1	1	1	1	1	0.18	0

From Tables 8 and 9, we observe that ICL with CBOW on single-relationship tasks performs better with squared loss compared to cross-entropy loss and with fewer demonstration examples. Also, ICL tends to deteriorate after a certain number of in-context demonstrations. As detailed in Appendix B, a smaller number of examples (e.g., zero) allows the model to produce the correct output instead of one of the r_i's. This is in contrast with transformer-based LLMs, which achieve better ICL performance as the number of demonstrations increases. On the other hand, ICL on dual-relationship tasks as described in Section 2.2 requires at least one demonstration example to distinguish between the two tasks.

D Proof of Theorem 2

Proof. We show that given a prompt of the form $c_{i_1} d_{i_1} \cdots c_{i_\ell} d_{i_\ell} c_{i_{\ell+1}}$ with distinct i_k's, a trained CBOW model is more likely to predict $d_{i_{\ell+1}}$ than $e_{i_{\ell+1}}$. If this is established, the other part of the theorem follows analogously. We now define

- $p_1 = p(c_i, d_j) = p(d_i, c_j) = p(d_i, d_j) = p(c_i, e_j) = p(e_i, c_j) = p(e_i, e_j)$ for any $i \neq j$;
- $p_2 = p(r_i, r_j)$ for any $i \neq j$;
- $p_3 = p(c_i, d_i) = p(d_i, c_i) = p(c_i, e_i) = p(e_i, c_i)$;
- $p_4 = p(d_i, r_j) = p(r_i, d_j) = p(e_i, r_j) = p(r_i, e_j)$ for any i, j;

where the equalities in the probabilities are a consequence of the data distribution. By direct calculation, we have $p_1 = \frac{1}{K(K-1)}$, $p_2 = \frac{(S-4)(S-5)}{L(L-1)}$, $p_3 = \frac{1}{K}$, and $p_4 = \frac{S-4}{KL}$, where each p_i is multiplied by $S(S-1) > 0$ (without loss of generalization) to make calculations easier. Moreover, it is easy to see that $p(c_i, r_j) = p(r_i, c_j) = 2p_4$ for any i, j and $p(c_i, c_j) = 2p_1$ for any $i \neq j$. Lastly, we define $a = (S-2)p_1$, $b = (S-2)p_2$, $c = (S-2)p_3$, $d = (S-2)p_4$, $e = 2(K-1)p_1 + p_3 + Lp_4$, and $f = 4Kp_4 + (L-1)p_2$.

The next step the proof is to use Lemma 1 in Appendix B to obtain the similarity matrix A. As previously, we denote a square matrix with α on the diagonal and β off the diagonal as $X_{\alpha,\beta}$, and a matrix with all entries γ as Y_γ. We then have

$$B = \begin{bmatrix} X_{0,2p_1} & X_{p_3,p_1} & X_{p_3,p_1} & Y_{2p_4} \\ X_{p_3,p_1} & X_{0,p_1} & Y_0 & Y_{p_4} \\ X_{p_3,p_1} & Y_0 & X_{0,p_1} & Y_{p_4} \\ Y_{2p_4} & Y_{p_4} & Y_{p_4} & X_{0,p_2} \end{bmatrix}$$

and

$$(S-2)B + C = \begin{bmatrix} X_{2e,2a} & X_{c,a} & X_{c,a} & Y_{2d} \\ X_{c,a} & X_{e,a} & Y_0 & Y_d \\ X_{c,a} & Y_0 & X_{e,a} & Y_d \\ Y_{2d} & Y_d & Y_d & X_{f,b} \end{bmatrix}. \tag{2}$$

Moreover, its inverse can be written as

$$((S-2)B + C)^{-1} = \begin{bmatrix} X_{q_2,q_1} & X_{q_3,q_1} & X_{q_3,q_1} & Y_{q_4} \\ X_{q_3,q_1} & X_{q_5,q_6} & X_{q_7,q_8} & Y_{q_4} \\ X_{q_3,q_1} & X_{q_7,q_8} & X_{q_5,q_6} & Y_{q_4} \\ Y_{q_4} & Y_{q_4} & Y_{q_4} & X_{q_9,q_{10}} \end{bmatrix}, \tag{3}$$

for some $q_1, q_2, \cdots, q_{10}$. Recall that our task is show that given context words $c_{i_1}, d_{i_1}, \cdots, c_{i_\ell}, d_{i_\ell} c_{i_{\ell+1}}$ with distinct i_k's, the center word is more likely to be $d_{i_{\ell+1}}$ than $e_{i_{\ell+1}}$. In other words, we need to establish that

$$\begin{aligned} & d_{i_{\ell+1}}^\top c_{i_1} + d_{i_{\ell+1}}^\top d_{i_1} + \cdots + d_{i_{\ell+1}}^\top c_{i_\ell} + d_{i_{\ell+1}}^\top d_{i_\ell} + d_{i_{\ell+1}}^\top c_{i_\ell+1} \\ & > e_{i_{\ell+1}}^\top c_{i_1} + e_{i_{\ell+1}}^\top d_{i_1} + \cdots + e_{i_{\ell+1}}^\top c_{i_\ell} + e_{i_{\ell+1}}^\top d_{i_\ell} + e_{i_{\ell+1}}^\top c_{i_\ell+1}, \end{aligned}$$

where $\epsilon^\top \delta$ indicates the similarity between the word ϵ in the center and the word δ in the context. This similarity can be obtained from the matrix $A = B((S-2)B + C)^{-1}$. By symmetry, the inequality reduces to $d_i^\top d_j > e_i^\top d_j$ for any $i \neq j$.

By computing the matrix A, we have

$$d_i^\top d_j = p_3 q_1 + p_1 q_3 + (K-2)p_1 q_1 + (K-2)p_1 q_6 + Lp_4 q_4 + p_1 q_5$$

and

$$e_i^\top d_j = p_3 q_1 + p_1 q_3 + (K-2)p_1 q_1 + (K-2)p_1 q_8 + p_1 q_7 + Lp_4 q_4.$$

Thus, our problem again reduces to showing $(K-2)q_6+q_5 > (K-2)q_8+q_7$ as $p_1 = \frac{1}{K(K-1)} > 0$. Upon multiplying (3) and (2) and equating the result with the identity matrix, we have the following equations:

$$a(K-1)q_1 + cq_3 + dLq_4 + eq_5 + a(K-1)q_6 = 1 \tag{4}$$
$$(c+a(K-2))q_1 + aq_3 + dLq_4 + aq_5 + (e+a(K-2))q_6 = 0 \tag{5}$$
$$a(K-1)q_1 + cq_3 + dLq_4 + eq_7 + a(K-1)q_8 = 0 \tag{6}$$
$$(c+a(K-2))q_1 + aq_3 + dLq_4 + aq_7 + (e+a(K-2))q_8 = 0. \tag{7}$$

Comparing (5) and (7) yields

$$a(((K-2)q_6+q_5) - ((K-2)q_8+q_7)) = e(q_8-q_6).$$

As $a = (S-2)p_1 > 0$ and $e = 2p_1(K-1)+p_3+p_4L > 0$, we now only need to show that $q_8 > q_6$. Comparing (4) and (6) as well as (5) and (7), we have

$$a(q_5-q_7) = (e+a(K-2))(q_8-q_6)$$
$$e(q_5-q_7) = a(K-1)(q_8-q_6)+1,$$

which reduces to $(q_8-q_6)(e^2+ae(K-2)-a^2(K-1)) = a$. The conclusion follows since $a > 0$ and

$$e^2+ae(K-2)-a^2(K-1) = (e-a)(e+a(K-1)) = \left(\frac{S-1}{K} - \frac{S-2}{K(K-1)}\right)(e+a(K-1)) > 0.$$

□

E Proof of Theorem 3

Proof. Consider the instance of predicting a from abc, i.e., $f(\{a, b\}, c)$. By the assumption on the data distribution, it is equally likely that the task is predicting b from bac. In this case, the corresponding function is also $f(\{a, b\}, c)$. Thus, the sum of the cross-entropy losses corresponding to these two tasks is lower bounded by $2 \log 2 > 0$. Also, it is easy to see that we cannot achieve perfect accuracy since the predictions for abc and bac must be the same.

We now show that it is possible to attain zero loss and perfect accuracy when the model includes positional embeddings, so that $\tilde{f}(\{(a, 1), (b, 2)\}, (c, 3)) \neq \tilde{f}(\{(b, 1), (a, 2)\}, (c, 3))$. As a special case, we consider a simplified version of the transformer architecture, where

$$\tilde{f}(\{(a, 1), (b, 2)\}, (c, 3)) = \frac{\sum_{k \in \{a,b,c\}} (x_k + p_1) \exp((x_k + p_1)^\top (x_c + p_3))}{\sum_{k \in \{a,b,c\}} \exp((x_k + p_1)^\top (x_c + p_3))}.$$

and

$$p(d \mid abc) \propto \exp\left(x_d^\top \tilde{f}(\{(a, 1), (b, 2)\}, (c, 3))\right).$$

for any token d. Here, x_i and p_j represent the embedding of token i and position j, respectively.

Let $p_1^\top p_3 = p$, $p_2^\top p_3 = q$, $p_3^\top p_3 = r$, $x_i^\top x_i = s$, $x_i^\top x_j = t$ for any $i \neq j$, $p_1^\top x_i = u$ for any i, $p_2^\top x_i = v$ for any i, and $p_3^\top x_i = w$ for any i. Note that this holds due to the assumed data generating process. We consider the following construction: $p_1 = b\mathbb{1}_{|V|}$, $p_2 = p_3 = \mathbb{1}_{|V|}$, and $x_i = ae_i$, where e_i is a zero vector with 1 on the i-th entry. This implies $p = b|V|$, $q = r = |V|$, $s = a^2$, $t = 0$, $u = ab$, and $v = w = a$.

By direct calculation, the cross-entropy loss of predicting a from abc is given by

$$-\log\left(\frac{\exp(\alpha_1 a^2)}{\exp(\alpha_1 a^2) + \exp(\alpha_2 a^2) + \exp(\alpha_3 a^2) + |V| - 3}\right),$$

where

$$\alpha_1 = \frac{\exp(ab + b|V|)}{\exp(ab + b|V|) + \exp(a + |V|) + \exp(a^2 + a + |V|)},$$

$$\alpha_2 = \frac{\exp(a + |V|)}{\exp(ab + b|V|) + \exp(a + |V|) + \exp(a^2 + a + |V|)},$$

$$\alpha_3 = \frac{\exp(a^2 + a + |V|)}{\exp(ab + b|V|) + \exp(a + |V|) + \exp(a^2 + a + |V|)}.$$

Letting $b = a^2$ and $a \to \infty$, it is easy to see that we can bring the cross-entropy loss arbitrarily close to zero. Consequently, we also have a perfect prediction accuracy. □

F Proof of Proposition 1

Proof. The intermediate representation of the first layer is given by $f_1(\{x_{i_1}\})$, $f_2(\{x_{i_1}\}, x_{i_2})\}$, $f_3(\{x_{i_1}, x_{i_2}\}, x_{i_3})$, and $f_4(\{x_{i_1}, x_{i_2}, x_{i_3}\}, x_{i_1})$, for some functions f_1, f_2, f_3, and f_4. To predict the last x_{i_1}, we use the third coordinate of the second layer representation, which is given by $t(x_{i_1} x_{i_2} x_{i_3}) := g_3\left(\{f_1(\{x_{i_1}\}), f_2(\{x_{i_1}\}, x_{i_2})\}, f_3(\{x_{i_1}, x_{i_2}\}, x_{i_3})\right)$, for some function g_3. It is easy to see that in general, $t(x_{i_1} x_{i_2} x_{i_3}) \neq t(x_{i_2} x_{i_1} x_{i_3})$. □

G Proof of Theorem 4

Proof. In the one-noisy scenario, each sentence takes one of the following forms: *nabacdc*, *anbacdc*, *abnacdc*, *abancdc*, *abacndc*, and *abacdnc*, where $n \in \mathcal{N}$. In order to achieve the minimum possible theoretical loss, we minimize each loss term separately. Concretely, the minimum loss of predicting the sixth token given the first five tokens is attained by the following rule:

- When the first five tokens do not contain any nuisance token, output a uniform probability vector over $\mathcal{N}$.
- Otherwise, output the conditional probability of $c[2]$ given x, where $(x, c[2]) \in S_1$. Here, x represents the last non-nuisance token.

Under this rule, the predicted output for any in-context example *abacd* is never c, since $c \notin \mathcal{N}$. In the block-noisy scenario, each sentence takes one of the following forms: $n_1n_2n_3abacdc$, $aban_1n_2n_3cdc$, and $abacdcn_1n_2n_3$, where $n_1, n_2, n_3 \in \mathcal{N}$. The minimum loss of predicting the ninth token given the first eight tokens is attained by the following rule:

- When the seventh token is not a nuisance token, output the seventh token with probability one.
- When the seventh token is a nuisance token, output a uniform probability vector over $\mathcal{N}$.

Under this rule, the predicted output for any in-context example *abacdcef* is *e*, resulting in perfect ICL accuracy.

□

H Proof of Theorem 5

Proof. Recall that each training sentence is of the form $x_{11}x_{12}x_{11}x_{21}x_{22}x_{21}\cdots x_{N1}x_{N2}x_{N1}$. Note that we can decompose the total loss $\mathcal{L}$ into $\mathcal{L}_1 + \mathcal{L}_2 + \cdots + \mathcal{L}_{3N}$, where $\mathcal{L}_g$ denotes the loss of predicting the g-th token given all the other previous tokens. As the $x_{i1}x_{i2}x_{i1}$ blocks are generated independently, the optimal loss should satisfy $\mathcal{L}_1 = \mathcal{L}_4 = \cdots = \mathcal{L}_{[3N-2]} = \mathcal{L}_{[1]}$, $\mathcal{L}_2 = \mathcal{L}_5 = \cdots = \mathcal{L}_{[3N-1]} = \mathcal{L}_{[2]}$, and $\mathcal{L}_3 = \mathcal{L}_6 = \cdots = \mathcal{L}_{[3N]} = \mathcal{L}_{[3]}$. Therefore, it is sufficient to minimize $\mathcal{L}_{[1]} + \mathcal{L}_{[2]} + \mathcal{L}_{[3]}$.

In order to achieve the minimum possible theoretical loss, we need to minimize $\mathcal{L}_{[1]}$, $\mathcal{L}_{[2]}$, and $\mathcal{L}_{[3]}$ separately. It is easy to see that $\mathcal{L}_{[1]}$ is minimized by outputting the marginal probability of $c[1]$, where $c \in S_1$. Similarly, $\mathcal{L}_{[2]}$ is minimized by outputting the conditional probability of $c[2]$ given x_{i_1}, where $(x_{i_1}, c[2]) \in S_1$. On the other hand, it is possible to achieve an $\mathcal{L}_{[3]}$ value of zero by outputting x_{i_1} with probability one.

Now, given an ICL prompt $\underline{x_{11}x_{12}x_{12}x_{21}x_{22}x_{22}} \cdots \underline{x_{\ell 1}x_{\ell 2}}$ where $\ell \leq N$, the trained model should predict $x_{\ell 1}$ with probability one since $\{c[1] \mid c \in S_2\} = \mathcal{V}$ and our ICL prompt corresponds to $\mathcal{L}_{[3]}$. This completes the proof. □

I Proof of Theorem 6

Proof. We proceed similarly as the proof of Theorem 5. Concretely, we separately minimize $\mathcal{L}_g$ for $g \in [2k+2]$, where $\mathcal{L}_g$ denotes the loss of predicting the g-th token given all the other previous tokens. It is easy to see that $\mathcal{L}_1$ is minimized by outputting a uniform probability vector over $a_{1:[I]}$, whereas $\mathcal{L}_h$ (for any $2 \leq h \leq 2k+1$) is minimized by outputting a uniform probability vector over $\mathcal{V}$. Moreover, it is possible to achieve an $\mathcal{L}_{2k+2}$ value of zero by outputting b_i with probability one.

From here, given an ICL prompt of the form $a_{i_1}b_{i_1}a_{i_2}b_{i_2}\cdots a_{i_\ell}$, the trained model should predict a uniform probability vector over $\mathcal{V}$ if $\ell \leq k$, and b_{i_1} if $\ell = k+1$. In all cases, the model does not predict b_{i_ℓ}, completing the proof. □

J Limitations and future work

This study has several limitations. Firstly, the experiments are conducted on a relatively small scale. However, they still provide sufficient evidence to support the theoretical findings. Secondly, the focus of this study is on specific types of in-context learning (ICL) tasks, as described in Section 1. Thirdly, the pre-training data considered in this work may not match the valid grammatical sentences that language models are usually trained on. Nonetheless, our co-occurrence results still apply to grammatical sentences, as the co-occurring pairs can appear naturally within them (e.g., "Beijing is the capital of China," or "the city of Beijing is located in China"). Lastly, real data sets are not utilized due to the lack of alignment with the study objectives.

Despite these limitations, we believe this work provides valuable understanding of the key factors enabling ICL to occur from training on unstructured natural language data, supported by both theoretical and empirical evidence from experiments involving prompting and synthetic data. Further analyses on other ICL tasks and their reliance on model architecture can be fruitful avenues for future work.

K Details of experiments and data sets

K.1 Architecture and implementation

All experiments utilize the Keras package in Python, employing the Adam optimizer [29] with a learning rate of 0.01. Early stopping is applied based on validation loss with a patience threshold of 5, utilizing a randomly selected subset representing 50% of the original data set. Each transformer layer uses two heads, as we empirically demonstrated that increasing the number of heads does not impact performance in our experiments. Each layer consists of the following components (in order): (1) Keras' multi-head causal self-attention block, with key_dim $=$ value_dim $=$ embed_dim$/2$; (2) Skip connection and layer normalization; (3) One hidden layer feed-forward network using the ReLU activation with dimension $= 2\times$ embed_dim; and (4) Skip connection and layer normalization.

K.2 Source and details of data sets

The *world_population.csv* data set, used for the experiments in Section 2, is obtained from Kaggle. According to the author, this data set is created from World Population Review.

The *us-state-capitals.csv* data set, used for the experiments in the beginning of Section 2, is obtained from this Github repository. Its source is unclear.

The *uscities.csv* data set, used for the experiments in the beginning of Section 2, is obtained from Simple Maps, with a CC 4.0 license.

K.3 Details of synthetic data used in experiments

Below we provide additional details regarding the synthetic data used in our experiments.

1. For experiments in Table 1, the training data consists of 50,000 sentences. In the clean version, sentences are generated uniformly as described in Section 2.1. In the corrupted version, sentences are generated in a similar manner, but each (c_i, d_i) pair is replaced by (c_i, r_j) or (d_i, r_j) with a probability of $1/4$ each. Test sentences are generated according to the setup in Theorem 1. Some examples are as follows:
   - Clean
     - Training: $c_1 d_1 r_1 r_2 r_3 r_4 r_5 r_6$ or $r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8$
     - Prompt: $c_1 d_1 c_2 d_2 c_3 d_3 c_4$?
   - Corrupted
     - Training: $c_1 r_1 r_2 r_3 r_4 r_5 r_6 r_7$ or $c_1 d_1 c_2 r_1 r_2 r_3 r_4 r_5$
     - Prompt: $c_1 d_1 c_2 d_2 c_3 d_3 c_4$?
2. For experiments in Table 2, the training data consists of 50,000 sentences. In the clean version, sentences are generated uniformly as described in Section 2.2. In the imbalanced and extreme versions, the 60 other words are divided into three categories: 20 for cd

sentences ($rcd.$), 20 for ce sentences ($rce.$), and 20 for both types ($r.$). In the imbalanced version, cd (ce) sentences are 4 times more likely to sample a cd (ce) word than a ce (cd) word. In the extreme version, cd (ce) sentences cannot contain any ce (cd) words. Test sentences are generated according to the setup in Theorem 2. Some examples are as follows:

- Clean examples
  - Training: $c_1d_1r_1r_2r_3r_4r_5r_6$ or $c_1e_1r_1r_2r_3r_4r_5r_6$
  - Prompt: $c_1d_1c_2d_2c_3d_3c_4?$ or $c_1e_1c_2e_2c_3e_3c_4?$
- Imbalance examples
  - Training: $c_1d_1rcd_1rcd_2rcd_3rce_4r_5r_6$ or $c_1e_1rcd_1r_2rce_3rce_4rce_5r_6$
  - Prompt: $c_1d_1c_2d_2c_3d_3c_4?$ or $c_1e_1c_2e_2c_3e_3c_4?$
- Extreme examples
  - Training: $c_1d_1rcd_1rcd_2rcd_3r_4r_5r_6$ or $c_1e_1r_1r_2r_3rce_4rce_5r_6$
  - Prompt: $c_1d_1c_2d_2c_3d_3c_4?$ or $c_1e_1c_2e_2c_3e_3c_4?$

3. Experiments in Table 3 follow the setup of experiments in Table 2, except that the pairs are now of the form (c_i, d_i) and (e_i, f_i) instead of (c_i, d_i) and (c_i, e_i).

4. For experiments in Section 2.4, the corpus generation process is as follows:
   - Randomly select 10 countries and obtain their capital cities and IOC codes.
   - Generate 30 sentences containing exactly one country-capital pair (3 for each country).
     Example: Paramaribo is the vibrant heart of Suriname.
   - Generate 30 sentences containing exactly one country-IOC pair (3 for each country).
     Example: Gabon (GAB) protects its diverse rainforests and wildlife.
   - Generate 30 sentences containing exactly one country without any pair.
     Example: The banking sector is central to Liechtenstein's prosperity.
   - Generate 60 sentences without any country, capital city, or IOC code.
     Example: Every country has its unique cultural identity and heritage.
   - Generate 810 sentences containing exactly two different country-capital pairs by concatenating sentences generated in Step 2.
     Example: The city of Dushanbe reflects Tajikistan's vibrant spirit. Roseau is the cultural tapestry of Dominica.
   - Generate 810 sentences containing exactly two different country-IOC pairs by concatenating sentences generated in Step 3.
     Example: Mayotte (MAY) features lush landscapes and peaks. Turkmenistan (TKM) features the fiery Darvaza Crater.

   The ICL prompts follow the form used in the country-capital city and US state-capital city experiments in the beginning of Section 2, with 1 to 5 in-context examples.

5. For experiments in Table 4, the training and test data consist of all sentences in the form $abca$, where a, b, and c are distinct. Each test sentence is different from any training sentence. In the first scenario (both), the first tokens of the training sentences cover the entire vocabulary. In the second scenario (either), each token can be the first token in either the training or test data, but not both.

6. For experiments in Table 5, the training data consists of 50,000 sentences generated uniformly as detailed in Section 3.1. The ICL prompt formats are also described in Section 3.1.

7. For experiments in Table 6, the training data consists of 50,000 sentences. In the clean scenario, the training data are of the form $abcadefd$ and $abcbdefe$, with ICL prompts as $\overline{abcadef}?$ and $\overline{abcbdef}?$. In the block-noisy scenario, the training data include sequences like $n_1n_2n_3n_4abcadefd$ and $abcbn_1n_2n_3n_4defe$, with ICL prompts as $\overline{abcadefdghi}?$ and $\overline{abcbdefeghi}?$.

8. For experiments in Table 7, the training data consists of 50,000 sentences generated uniformly according to the processes in Sections 4.1 and 4.2. The ICL prompt formats are also described in the same subsections.

9. For experiments in Section 4.3, the corpus generation process is as follows:
   - Randomly select 10 countries and obtain their capital cities and IOC codes.

- Generate 130 sentences containing exactly one country-capital pair (13 for each country).
  Example: Paramaribo stands as capital of Suriname.
- Generate 30 sentences containing exactly one country without any pair.
  Example: The banking sector is central to Liechtenstein's prosperity.
- Generate 60 sentences without any country, capital city, or IOC code.
  Example: Every country has its unique cultural identity and heritage.
- Generate 1,000 sentences containing exactly two different country-capital pairs by concatenating sentences generated in Step 2.
  Example: Brazil functions as heart of Brasilia. Turkmenistan operates as center for Ashgabat.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: Abstract and introduction (Section 1) clearly summarize the paper's motivation (relevance), scope, and contributions. A summary of contributions at the bottom of Section 1 is provided to re-iterate the contributions of this paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: Limitations of this work are included in Appendix J.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes] .

Justification: Assumptions of each theorem and lemma are clearly mentioned in the theorem/lemma statement. Correct and complete proofs of all theoretical results are provided in the appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes] .

Justification: Codes for each experiment are provided as a .zip file in the submission. Experimental details are also provided throughout the paper and in Appendix K.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] .

   Justification: Open access to all codes is provided (one folder for each result). File names are self-explanatory.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes] .

   Justification: Enough details are provided throughout the paper and in Appendix K. See code for more details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No] .

   Justification: We only repeat each experiment 10 times due to computational constraints. However, most experimental results are backed up by theoretical analyses.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No] .

   Justification: We do not include details on the type amd amount of compute required to run each experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes] .

   Justification: We have reviewed the NeurIPS Code of Ethics and concluded that this research conforms with it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA] .

    Justification: There is no societal impact as this work is foundational.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: We believe this paper poses no such risks as we do not train large-scale language models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes] .

    Justification: The existing models used in this paper (e.g., LLaMA) are properly credited. The sources of the data sets used in our experiments are mentioned in Appendix K.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA] .

    Justification: This paper does not include any new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.