Position: Do pretrained Transformers Learn In-Context by Gradient Descent?

Lingfeng Shen [*1] **Aayush Mishra** [*1] **Daniel Khashabi** [1]

Abstract

The emergence of In-Context Learning (ICL) in LLMs remains a remarkable phenomenon that is partially understood. To explain ICL, recent studies have created theoretical connections to Gradient Descent (GD). We ask, do such connections hold up in actual pre-trained language models? We highlight the limiting assumptions in prior works that make their setup considerably different from the practical setup in which language models are trained. For example, their experimental verification uses *ICL objective* (training models explicitly for ICL), which differs from the emergent ICL in the wild. Furthermore, the theoretical hand-constructed weights used in these studies have properties that don't match those of real LLMs. We also look for evidence in real models. We observe that ICL and GD have different sensitivity to the order in which they observe demonstrations. Finally, we probe and compare the ICL vs. GD hypothesis in a natural setting. We conduct comprehensive empirical analyses on language models pre-trained on natural data (LLaMa-7B). Our comparisons of three performance metrics highlight the inconsistent behavior of ICL and GD as a function of various factors such as datasets, models, and the number of demonstrations. We observe that ICL and GD modify the output distribution of language models differently. These results indicate that *the equivalence between ICL and GD remains an open hypothesis* and calls for further studies.

1. Introduction

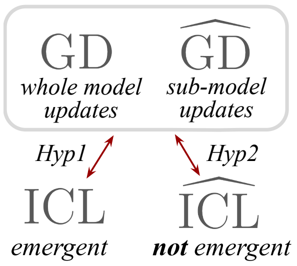
In-Context Learning (ICL) is an emergent behavior in Large Language Models (LLMs), which allows them to recognize patterns among demonstrations provided as prompts and extend these patterns to similar tasks (Brown et al., 2020). This fascinating on-the-fly learning behavior has motivated ample studies to better of understand its dynamics.

In particular, a notable line of work tries to explain ICL via Gradient Descent (GD) (Garg et al., 2022; Zhang et al., 2023). This connection is interesting because GD has been around for decades and is well-understood, while ICL is a recent phenomenon that has emerged somewhat surprisingly (Wei et al., 2022), and is not fully understood. Therefore, a solid formal bridge between the two approaches would be an exciting finding as it can open new doors for understanding ICL.

> **Hypothesis 1.** For any Transformer weights resulting from self-supervised pretraining and for any well-defined task, ICL is algorithmically equivalent to GD (whole model or sub-model).

In this work, we revisit the hypothesis on the equivalence of ICL and GD, i.e., whether these two approaches to "learning" are functionally equivalent. Consider hypothesis 1 that defines a *universal* notion of equivalence between the ICL and GD. It defines equivalence as a property that must hold for *any* Transformer model with parameters that *emerge* naturally from pretraining on massive unlabeled data (Brown et al., 2020), and is applicable for *any* choice of well-defined tasks (Srivastava et al., 2023). For example, (Dai et al., 2023) claims that ICL is equivalent to implicit finetuning.

> **Hypothesis 2.** For a given well-defined task, there exist Transformer weights such that $\widehat{\text{ICL}}$ is algorithmically equivalent to GD (whole model or sub-model).



However, other recent works have focused on a different claim outlined in hypothesis 2, which focuses on in-context learning behavior that is **not emergent** (denoted as $\widehat{\text{ICL}}$). This deviates from hypothesis 1 in the family of models (differences in training setups) and family of tasks, as we will see in detail in §3. This hypothesis articulates a tangential target: being able to *simulate* GD

[*]Equal contribution [1]Johns Hopkins University, Baltimore MD. Correspondence to: Aayush Mishra <amishr24@jhu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

on a given task with *some* (trained or hand-constructed) Transformer weights. This is mainly concerned with the **expressivity** of Transformer architecture (Merrill et al., 2022; Chiang et al., 2023), ignoring how it may emerge from pre-training. A few notable works use this hypothesis to provide a theoretical argument for the ICL≈GD claim. Specifically, (Akyürek et al., 2022; von Oswald et al., 2023) show (via a different set of arguments) that Transformer-based architectures (Vaswani et al., 2017), for appropriate choices of parameters, can process their in-context observations in a way that is equivalent to running gradient updates on an *implicit sub-model*'s parameters using the same demonstrations.

These claims are made under strong assumptions, which raises the question of whether they hold in practice. Specifically, **do the recent results focusing on hypothesis 2 provide any (even partial) evidence for hypothesis 1?** Although these works highlight interesting abilities of the Transformer architecture, their claims about the equivalence between ICL and GD are *too strong* for real-world models.

We divide our study into three parts. In the first part (§3), we show that previous works that study the ICL≈GD hypothesis make assumptions that are hard to justify in the real world (hypothesis 2). Then, we use order-sensitivity as an argument against the equivalence between ICL and GD (§4). Finally, we put these claimed equivalences to the test (§5) by presenting a comprehensive empirical study. Our experiments reveal that ICL operates and performs differently from GD (fine-tuning the whole model or intuitive sub-models) on real-world language models across a variety of model sizes, datasets and the number of demonstrations.

In summary,

1. We provide arguments against existing theories of equivalence between ICL and GD, highlighting the gap between their experimental setup and real-world transformers.
2. We empirically evaluate the equivalence between ICL and GD in the real-world setting using a variety of metrics and find that the two function quite differently.
3. We call for more nuanced studies that maintain parallels with real-world LLMs so their inferences about ICL can be practically useful.

2. Background

We start with our problem setting (§2.1). We use "sampling" to emphasize *a priori unknown problem parameters*. Specifically, our computational setup consists of sampling (choosing) a learning problem (task) and correspondingly sampling (training) a pretrained model. We then cover the two learning setups studied for equivalence (§2.2), followed by the treatment of ICL≈GD hypothesis in recent literature.

2.1. Sampling tasks and models

Sampling from the space of well-defined tasks. Consider a family of functions (tasks) $\mathcal{F}$ such that each $(f : \mathcal{X} \to \mathcal{Y}) \in \mathcal{F}$, maps inputs in the domain $\mathcal{X}$ to the domain $\mathcal{Y}$. A particular function $f \in \mathcal{F}$ elicits a sampling process $x \overset{f}{\sim} \mathcal{X}$ which samples input from $\mathcal{X}$ such that they are compatible with f. For example, in natural language, $\mathcal{F}$ defines the space of all tasks that involve mapping from language input to language output, like sentence completion, summarization, QA, translation, etc. However, each task f (e.g., translating English to French) would require specific inputs (English and not, say, German) pertinent to the task. The goal is to find models that learn (imitate) f by conditioning on a set of examples $S^f = \left\{ S_i^f = (x_i, f(x_i)) \middle| f \sim \mathcal{F}, x_i \overset{f}{\sim} \mathcal{X} \right\}$. The model's competence is then evaluated using a test set $S^f_{\text{test}} = \{(x_i^t, f(x_i^t))\}$, which is disjoint from S^f. During the evaluation, only the inputs in S^f_{test} (which we denote as X^f_{test}) are shown to the model.

Sampling from the space of pretrained models. LLMs like GPT and LLaMa (Brown et al., 2020; Touvron et al., 2023) are pretrained using the Causal Language Modelling (CLM) objective (Radford et al., 2019) which is more commonly understood as *next-word prediction* objective (Liu et al., 2018). This process of pretraining elicits a family of models $\mathcal{M}$ depending primarily on the *data distribution and characteristics of sequences*, and additionally on the choice of architectures, initializations, etc. Formally, we denote this model M_{Θ_0} with pretrained weights Θ_0, which is one model sampled from a much larger space of low perplexity pretrained models: $M_{\Theta_0} \sim \mathcal{M}$.

2.2. Standard Learning Setups

We review the standard treatment of ICL and GD and introduce the relevant notation.

In-context learning (ICL). We follow the dominant definition of In-context Learning (ICL) (Brown et al., 2020), which involves conditioning pretrained LLMs with a handful of examples of task f. Given these demonstrations, we want the LLM to perform f on new inputs. Formally, given demonstrations $S^f = \{S_i^f\}_{i=1}^N$ and a test input $x_i^t \in X_{\text{test}}$, the model M_{Θ_0} generates a label y_t when presented as $M_{\Theta_0}(S_1^f \circ S_2^f \circ ...S_N^f \circ x_i^t)$ or $M_{\Theta_0}(x_1 \circ f(x_1) \circ x_2 \circ f(x_2)...x_N \circ f(x_N) \circ x_i^t)$, where $\circ$ is a delimiter like newline which separates the instances. M_{Θ_0} produces a confidence distribution $\in \mathbb{R}^{|V|}$ over the vocabulary set V.

Gradient Descent (GD). Gradient Descent is an iterative numerical optimization algorithm used to minimize a given objective with respect to model parameters. Given a model with initial parameters Θ_0 and a differentiable loss function $\mathcal{J} \in \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, the algorithm updates the parameters toward the negative gradient $\nabla_{\Theta_0} \mathcal{J}$. GD is a standard optimizer used to train neural networks including LLMs. Although there are variants, like SGD and Adam, that work well in practice, we focus our study on vanilla GD, which calculates the gradients and takes a step (learning rate η) of fixed size. In the context of learning from a set of demonstrations, pretrained models $M_{\Theta_0} \sim \mathcal{M}$ are fine-tuned on a particular task f using GD by updating model parameters. Formally, parameter updates on the model M_{Θ_0} are performed for some epochs using the available demonstrations $S^f = \{S_i^f = (x_i, f(x_i))\}_{i=1}^N$ as follows:

$$\Theta_1 = \Theta_0 - \eta \nabla_\Theta \left(\frac{1}{N} \sum_{\substack{(x_i, f(x_i)) \\ \in S^f}} \mathcal{J}\left(M_{\Theta_0}(x_i), f(x_i)\right) \right) \tag{1}$$

After this process, the model is expected to perform this task given a new test sample *directly as input*: $M_{\Theta_1}(x_i^t)$.

3. The limiting assumptions in the study of ICL≈GD hypothesis

We highlight how recent studies drift from these conventional definitions of ICL and GD (§2.2) to support another form of equivalence. Specifically, they put restrictive assumptions on both the space of models $\mathcal{M}$ and the space of tasks $\mathcal{F}$ when training Transformers. Additionally, they impose impractical assumptions on model weights needed to prove their notion of equivalence between ICL and GD. We discuss why these deviations from real practice are non-trivial and offer little support for the equivalence between ICL and GD in practical settings. Fig.1 encapsulates the theme of our arguments discussed in detail next.

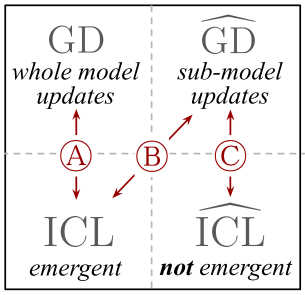


Figure 1: Ⓒ is discussed in §3. Ⓐ, Ⓑ in §4, §5;

3.1. Real LLMs are <u>not</u> pretrained with ICL objective

The widely-known ability of ICL *emerges* in pre-trained models ($\mathcal{M}$) that are obtained by training on CLM objective with natural language text as described in §2.1. Sequences in the pretraining corpus of natural language have a complicated relationship with the family of tasks $\mathcal{F}$ that they can perform using ICL. Understanding this relationship is an active area of research (cf. §6). However, we know that the pretraining corpus does not exclusively and explicitly contain sequences pertinent to $\mathcal{F}$. We refer to this training of Transformers with "natural" data (not necessarily natural language), which *does not* explicitly train it to perform ICL, as training with the *CLM objective*.

However, recent works use a different set of objectives. In Akyürek et al. (2022); von Oswald et al. (2023); Garg et al. (2022), the models are trained using the *ICL objective*:

$$\arg\min_\Theta \mathbb{E}_{\substack{f \sim \widehat{\mathcal{F}} \\ x_i \overset{f}{\sim} \mathcal{X}}} \left[\mathcal{L}\left(f(x_i), M_\Theta(x_1 \circ f(x_1) \circ x_2 \circ f(x_2) \ldots \circ x_i) \right) \right]. \tag{2}$$

This deviates from the real settings in at least two aspects:

Changing the space of tasks. This objective trains the model on the same restricted task distribution that it is tested on via ICL. We call this $\widehat{\text{ICL}}$, or the ability to perform ICL by training on *ICL objective* (cf. Figure 1) and the corresponding family of tasks $\widehat{\mathcal{F}}$. For example, if the target task to learn is linear regression, the model is trained on the sequence of linear regression instances. Therefore, this setup does not necessarily capture the essence of how ICL *emerges* in LLMs, which are not trained to perform ICL on a family of tasks.

Changing the space of models. Moreover, optimizing for this objective elicits a family of models $\widehat{\mathcal{M}}$ that is embedded with the inductive bias of expecting a constant structure in the sequence: a series of (x, y) pairs followed with a query input. Combined with the training on sequences specifically related to a restricted family of tasks $\widehat{\mathcal{F}}$, this space of models has different characteristics from the space of models $\mathcal{M}$ defined in §2.1.

The relationship between these sets of models is neither clear nor discussed in these recent works. Therefore, these works essentially equate $\widehat{\text{ICL}}$ with $\widehat{\text{GD}}$ (Ⓒ in Figure 1). Although restricted to a stricter family of tasks like Linear Regression is reasonable for analysis, it is important to discuss these distinctions between the setups. Using the term *Transformers* to refer to both these spaces of models and using the term ICL for $\widehat{\text{ICL}}$ are both misleading.

3.2. <u>Hand-constructed</u> weights and their limits

In this section, we analyze the weight matrices constructed by von Oswald et al. (2023) and Akyürek et al. (2022). As no method is provided to arrive at these weights by training, we place these hand-constructed weights under the umbrella of $\widehat{\text{ICL}}$. Next, we show how they are hard to justify for real-world language models (e.g., LLaMa-7B).

We first re-write the weight matrices of Transformers constructed by von Oswald et al. (2023). Their proposition states that given a reference linear model W, there exist key, query, value, and projection matrices (W_K, W_Q, W_V, P) of a Transformer such that a forward pass in that Transformer is identical to a gradient descent step on W, i.e., $e_j \leftarrow (x_j, y_j) + (0, -\Delta W x_j) = (x_i, y_i) + PVK^T q_j$.

The weight update ΔW is calculated by the mean squared error loss on the in-context samples as $\Delta W = -\eta \nabla_W L(W) = -\frac{\eta}{N}\sum_{i=1}^{N}(Wx_i - y_i)x_i^T$.

They construct $W_K = W_Q = \begin{pmatrix} I_x & 0 \\ 0 & 0 \end{pmatrix}, W_V = \begin{pmatrix} 0 & 0 \\ W_0 & -I_y \end{pmatrix}$ and $P = \frac{\eta}{N}I$, where I_x, I_y and I are identity matrices of size N_x, N_y and $N_x + N_y$ respectively. Using these matrices, they achieve the dynamics of a gradient step in the forward pass of a Linear Self Attention Layer (without softmax). The construction by Akyürek et al. (2022) is more complex and requires multiple steps to simulate one step of GD on one in-context sample. However, the construction is similar in that it is similarly sparse (see section C.4 in Akyürek et al. (2022)'s appendix). These constructions raise multiple concerns about their scaling to real-world models.

How does the model arrive at the correct P? In the construction by von Oswald et al. (2023), P is trivially assigned the value $\frac{\eta}{N}I$ which would change with the number of in-context samples. There is no insight into how a Transformer model would arrive at this information and how this formation behaves without any in-context samples. An edge case is $N = 0$ (no demonstrations), which surprisingly makes terms in P go to infinity.

Are LLM weights this sparse? The weight construction by von Oswald et al. (2023) has a lot of extremely sparse weight matrices. To be precise, W_K and W_Q would be matrices with N_x terms equal to 1 in the top left of the diagonal with the rest of $(N_x + N_y)^2 - N_x$ terms equal to zero. For LLaMa, the embedding size of the token vector, $N_x = N_y = 4096$. This means that the sparsity ratio (SR) in the weight matrices should be $\frac{((N_x+N_y)^2 - N_x)}{(N_x+N_y)^2} > 99.99\%$. The sparsity ratio in W_V should be close to $\approx 75\%$ if we assume each element in W_0 to be non-zero. In practice, the sparsity ratio is much lower for real-world models like LLaMa and GPT-J. As precisely 0 values for weights are unlikely, we measured the sparsity ratio in W_K, W_Q, and W_V by measuring weights less than a threshold (δ). Figure 2 shows the average sparsity value across layers for LLaMa. Overall, real-world pretrained Transformers have a much lower sparsity ratio than the assumptions.

How does ICL evolve during training? From the given constructions, models need to arrive at very specific weights to be able to perform gradient descent on in-context samples, but in practice, we observe models develop, retain, and improve this ability over time in training when the parameters change significantly (A detailed experimental setup is deferred to Appendix B). In Figure 3, we look at how the ability to perform ICL evolves compared with how the model parameters change over time (for each check-pointed GPT-J model). We measure the average parameter changes across all layers across W_K, W_Q, and W_V. This reveals that real Transformers do not settle on one set of weights (as required by previous works for performing GD) but continue to evolve throughout training. Although this result is an average over all the weights, certain groups of parameters (as constructed in previous works) are unlikely to remain constant throughout training. Therefore, ICL emerges in real LLMs, not just for a single choice of parameters but a family of parameters. Hence, **to prove the equivalence between GD and ICL, showing it for a single choice of parameters is not enough.**

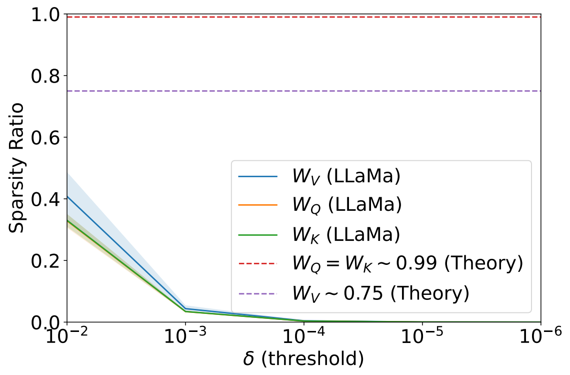


Figure 2: We show that the sparsity ratio in LLaMA (averaged across layers with standard deviation shown with shade) is much less than required by previous works to implement GD. More plots in Appendix C.

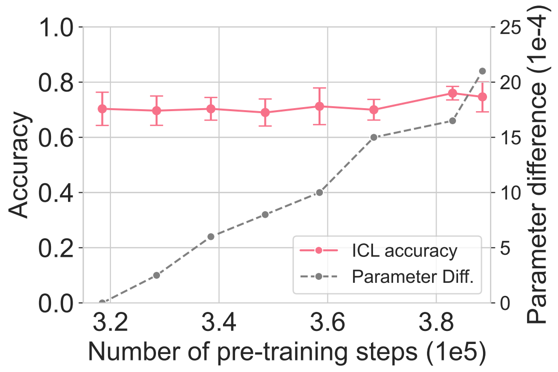


Figure 3: GPT-J's ability to do ICL (on AGNews) does not change much over a time cross-section of training while the parameters change steadily.

4. ICL is likely not equivalent to <u>order-stable</u> algorithms

While we established some limiting assumptions in previous studies, it remains unclear whether ICL≈GD hypothesis is actually invalid for real LLMs (Ⓐ or Ⓑ in Figure 1). For two algorithms to be equivalent, they must also have the *same functional behavior*. Namely, they should respond identically to the changes in the ordering of the instances. In this section, we discuss the discrepant sensitivity of ICL and GD to the order in which they process training instances (demonstrations).

Let's begin with the definition of algorithmic equivalence.

Definition 1 (Algorithmic equivalence to ICL)**.** *Consider an optimization algorithm $\mathcal{A}$ that modifies a pretrained model $M_{\Theta_0} \in \mathcal{M}$, using demonstrations $S = \{(x_i, f(x_i)\}_{i=1}^{N}$ of a well defined task $f \sim \mathcal{F}$, i.e., $\Theta_S \leftarrow \mathcal{A}(S, M_{\Theta_0})$. We call $\mathcal{A}$ "equivalent" to ICL if and only if the following holds:*

$$M_{\Theta_0}(S_1 \circ S_2 \circ ... S_N \circ x^t) = M_{\Theta_S}(x^t) \quad \forall\, x_i, x^t \overset{f}{\sim} \mathcal{X}. \quad (3)$$

The following theorem establishes the equivalence of order sensitivity between ICL and any algorithm $\mathcal{A}$ equivalent to it:

Theorem 1 (Algorithmic equivalence implies the same order sensitivity)**.** *Given a pretrained model $M_{\Theta_0} \in \mathcal{M}$, an algorithm $\mathcal{A}$ equivalent to ICL, and demonstrations $S = \{(x_i, f(x_i)\}_{i=1}^{N}$ of a well defined task $f \sim \mathcal{F}$, let σ_A, σ_B denote two orders of elements in S, such that $\Theta_{\sigma_A} \leftarrow \mathcal{A}(\sigma_A, M_{\Theta_0})$ and $\Theta_{\sigma_B} \leftarrow \mathcal{A}(\sigma_B, M_{\Theta_0})$. Then, for $\forall\, x^t \overset{f}{\sim} \mathcal{X}$, we have*

$$\underbrace{M_{\Theta_0}(\sigma_A \circ x^t) - M_{\Theta_0}(\sigma_B \circ x^t)}_{\textit{The order sensitivity of ICL}} = \underbrace{M_{\Theta_{\sigma_A}}(x^t) - M_{\Theta_{\sigma_B}}(x^t)}_{\textit{The order sensitivity of algorithm } \mathcal{A}},$$

Proof. The proof trivially follows from definition 1. We know that, $\forall\, x^t \overset{f}{\sim} \mathcal{X}$ we have:

$$M_{\Theta_0}(\sigma_A \circ x^t) = M_{\Theta_{\sigma_A}}(x^t)$$
$$M_{\Theta_0}(\sigma_B \circ x^t) = M_{\Theta_{\sigma_B}}(x^t).$$

Simply subtracting these two terms proves the theorem. □

4.1. ICL is likely not GD based on order inconsistency

Let's assume that GD is equivalent to ICL (arrow Ⓐ in Figure 1). We show that this assumption leads to a contradiction due to their inconsistent order sensitivity.

GD is order-stable. We know that GD is performed on a batch of samples from the training distribution, as seen in Equation 1. It does not matter which order the samples are presented. GD calculates the gradient using the average loss

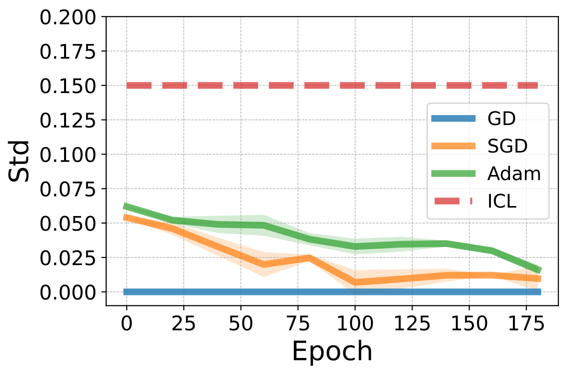


Figure 4: Order Sensitivity (standard deviation in output probabilities over the vocabulary) of ICL and GD (and its variants SGD and Adam) as measured on the LLaMa-7B on AGNews. The std is taken across 10 different orders of 8 ICL demos. More results are deferred to Appendix A.

across all samples and is therefore agnostic of the order in which they are calculated. With respect to theorem 1, if $\mathcal{A}$ = GD, $M_{\Theta_{\sigma_A}} = M_{\Theta_{\sigma_B}}$ or $M_{\Theta_{\sigma_A}}(x^t) - M_{\Theta_{\sigma_B}}(x^t) = 0$.

ICL and GD show different order-sensitivity. For ICL to be equivalent to any order-stable algorithm like GD, it must also be order-stable. However, previous research (Lu et al., 2022; Hahn & Goyal, 2023) has demonstrated that ICL is highly sensitive to the order of in-context samples. This is also easy to see because decoder-only Transformers exhibiting ICL only predict a token based on what they have seen before in the input. A different order of samples would change the behavior of the model. Therefore, ICL can not be equivalent to GD (arrow Ⓐ in Figure 1) as claimed by (Dai et al., 2023). These conclusions may change upon notable technological shifts (e.g., the architecture of LLMs). We also empirically verify this phenomenon by comparing the output distributions produced by ICL and GD (Figure 4). Details are deferred to Appendix A.

4.2. ICL is likely not $\widehat{\text{GD}}$ based on order inconsistency

Gradient Descent on *implicit sub-model* ($\widehat{\text{GD}}$). (Akyürek et al., 2022; von Oswald et al., 2023) also hypothesize the existence of *implicit sub-models* inside the weights of Transformer models. These sub-models (parameterized to perform linear regression) are constructed into the weights of the Transformer. When the Transformer is presented with in-context samples, it can simulate steps of gradient descent on the regression loss (using these samples) with respect to the sub-model parameters. Formally, for a sub-model with weights W_0, the Transformer model $M_{\Theta_0} = M_{\Theta_0 \backslash W_0, W_0}$ with fixed parameters $(\Theta_0 \backslash W_0)$ would optimize the weights of the inbuilt implicit sub-model (W_0) when presented with in-context samples and make its final prediction using updated weights (W_1). We refer to this version of GD as $\widehat{\text{GD}}$.

Now we define the equivalence of ICL to an algorithm that updates the implicit model only.

Definition 2. *Consider an optimization algorithm $\mathcal{A}$ that modifies the implicit sub-model weights W_0 of a pre-trained model $M_{\Theta_0} \in \mathcal{M}$, using demonstrations $S = \{(x_i, f(x_i))\}_{i=1}^N$ of a well defined task $f \sim \mathcal{F}$, i.e., $W_S \leftarrow \mathcal{A}(S, W_0)$. We call $\mathcal{A}$ "equivalent" to ICL if and only if the following holds, given $\forall\, x_i, x^t \overset{f}{\sim} \mathcal{X}$:*

$$M_{\Theta_0 \setminus W_0, W_0}(S_1 \circ S_2 \circ ... S_N \circ x^t) = M_{\Theta_S \setminus W_S, W_S}(x^t) \quad (4)$$

and $\Theta_0 \setminus W_0 = \Theta_S \setminus W_S$, i.e., the pretrained model only updates by the sub-models weights.

When the model with implicit sub-model weights W_0 is provided with in-context examples, it arrives at updated weights W_S using $\mathcal{A}$ without changing any other weights. This is equivalent to when the model starts with sub-model weights W_S and is provided no in-context examples, so no update happens on the weights via $\mathcal{A}$. Now, based on Definition 2 and Theorem 1, the following corollary about the equivalence of order sensitivity between ICL and an equivalent algorithm $\mathcal{A}$ also holds:

Corollary 1. *For a pretrained model $M_{\Theta_0} \in \mathcal{M}$, an algorithm $\mathcal{A}$ equivalent to ICL (according to definition 2) and two orders σ_A, σ_B of elements in the demonstration set S, $\forall\, x^t \overset{f}{\sim} \mathcal{X}$,*

$$\begin{aligned} &M_{\Theta_0 \setminus W_0, W_0}(\sigma_A \circ x^t) - M_{\Theta_0 \setminus W_0, W_0}(\sigma_B \circ x^t) \\ &= M_{\Theta_{\sigma_A} \setminus W_{\sigma_A}, W_{\sigma_A}}(x^t) - M_{\Theta_{\sigma_B} \setminus W_{\sigma_B}, W_{\sigma_B}}(x^t) \end{aligned} \quad (5)$$

ICL and $\widehat{\textbf{GD}}$ show different order-sensitivity. Let's assume that $\widehat{\text{GD}}$ is equivalent to ICL (arrow Ⓑ in Figure 1) according to definition 2. According to the same argument as in §4.1, $W_{\sigma_A} = W_{\sigma_B}$ or $\Theta_{\sigma_A} \setminus W_{\sigma_A}, W_{\sigma_A} = \Theta_{\sigma_B} \setminus W_{\sigma_B}, W_{\sigma_B}$ or $M_{\Theta_{\sigma_A} \setminus W_{\sigma_A}, W_{\sigma_A}}(x^t) - M_{\Theta_{\sigma_B} \setminus W_{\sigma_B}, W_{\sigma_B}}(x^t) = 0$. This again implies that for ICL to be equivalent to $\widehat{\text{GD}}$, it must be order-stable. Again, empirical evidence in today's LLMs shows that ICL is not order-stable and hence not equivalent to $\widehat{\text{GD}}$ (arrow Ⓑ in Figure 1). These conclusions may change in future.

What about variants of GD? We note that the construction of Akyürek et al. (2022) allows for order sensitivity in GD as the update is performed on samples one by one instead of the batch update performed by von Oswald et al. (2023). Although it is unclear which order is used to perform this update, we compared the order-sensitivity of ICL with SGD and Adam (Figure 4) and found that ICL is still significantly more sensitive to order than SGD/Adam. Therefore, it is unlikely that ICL is equivalent to even variants of GD. We provide more order-sensitivity results in Appendix A.

5. Empirical evalutation of ICL vs. GD/$\widehat{\textbf{GD}}$ in large pre-trained language models

This section provides an empirical evaluation of ICL≈GD equivalence in realistic settings. Specifically, we take a language model pretrained on natural data and use it with ICL demos to get ICL outputs. Then, we use the same demos to fine-tune the model using GD and $\widehat{\text{GD}}$, and get their respective output (without ICL demos). Next, we compare these outputs on various metrics to see how well ICL and GD/$\widehat{\text{GD}}$ align in practice.

5.1. Experimental settings

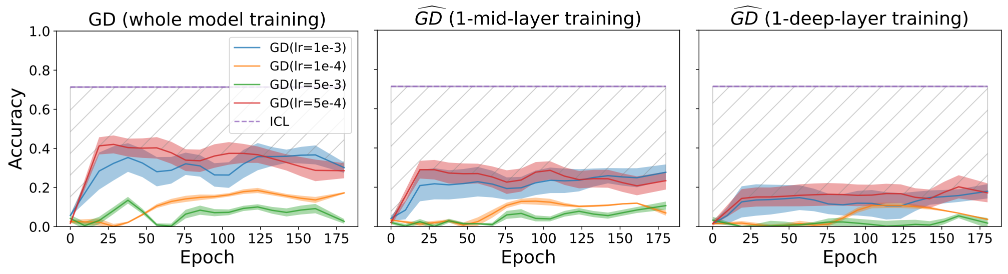
Model and benchmarks. We choose LLaMa (7B) (Touvron et al., 2023) as our primary model for evaluation. Our model-size comparative studies use the GPT family of models (as discussed later §5.2). For benchmarking, we select the following datasets: AGNews (Zhang et al., 2015), CB (De Marneffe et al., 2019), SST-2 (Socher et al., 2013), and RTE (Dagan et al., 2005).

Experimental setup. We evaluate ICL with varying demonstration sizes $N \in \{1, 2, 4, 8\}$ and for GD, we fine-tune the models with the same corresponding ICL demonstrations, experimenting with a variety of learning rates {1e-4, 5e-4, 1e-5, 5e-5} over 200 epochs, which ensures the convergence of model. Specifically, the objective function of GD is $\mathcal{J} = \sum_{(x,y)\in S} \mathcal{L}_{\text{clm}}(y; x)$, where $\mathcal{L}_{\text{clm}}(y; x)$ is the CLM loss of y, given x as the prefix. It is noteworthy that we only use gradients of the label and not the whole prefix to update the model. This is done to keep settings similar to the existing formalisms around ICL≈GD equivalence, where only output loss is calculated.

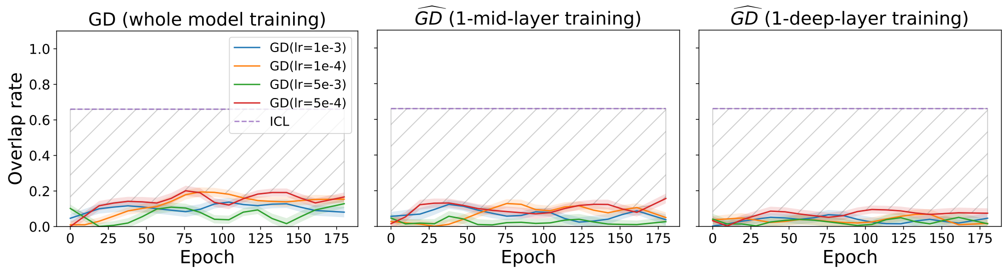
For $\widehat{\text{GD}}$, it is not trivial to identify the *implicit* sub-model as described in §4.2. Moreover, it is computationally infeasible to experiment on all possible subsets of parameters to identify the sub-model. Therefore, we use the hypotheses in Akyürek et al. (2022); von Oswald et al. (2023), to experiment with intuitive subsets. According to von Oswald et al. (2023) the *implicit* model lies in W_V of the Transformer while the probing experiments in Akyürek et al. (2022) suggest that this iterative optimization happens in top layers of the Transformers. This guides us to choose two intuitive subsets to simulate $\widehat{\text{GD}}$:

1. W_V of a single deep layer.
2. W_V of a single middle layer (for comparison).

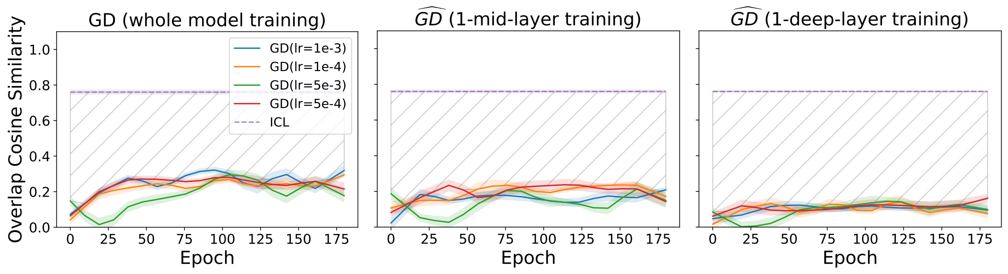
Overall, we compare ICL to GD, $\widehat{\text{GD}}$ (mid), and $\widehat{\text{GD}}$ (deep). Exact details about this setup are deferred to Appendix E.



(a) *Accuracy* comparison of ICL and GD variants.



(b) *Token Overlap* of ICL with GD variants.



(c) *Overlap Cosine Similarity* of ICL with GD variants.

Figure 5: Comparison of ICL and GD/$\widehat{\text{GD}}$ on our three metrics for the AGNews dataset (with 4 ICL demos). ICL lines in *Token Overlap* and *Overlap Cosine Similarity* are calculated between two different ICL output distributions (with different order of demonstrations in the prompt). A substantial gap between ICL and GD is highlighted by the gray diagonal lines.

Evaluation metrics. Previous works often use standard performance metrics (accuracy and loss) based on the token with the maximum probability from *label set* $\mathcal{Y}$ (Srivastava et al., 2023; Wei et al., 2021). We argue that these metrics do not paint the whole picture. Even if two sorting algorithms reach the same result, their dynamics may differ. For this purpose, we propose to look at *relative uplifting* of tokens in the output distribution. This nuanced analysis presents finer-grained information. A match/mismatch at the distributional level sheds more light on the dynamics of the algorithm. Therefore, we use the following metrics for analysis.

Accuracy: It is calculated using the target labels and predicted tokens with highest probability mass from the whole vocabulary V (rather than just the label set $\mathcal{Y}$) as it better evaluates the model's understanding of the task. It is defined as $\frac{1}{|S_{\text{test}}|}\sum_{(x_i^t,y_i^t)\in S_{\text{test}}} \mathbf{1}\{y_i^t = \arg\max M(C \circ x_i^t)\}$, where M is the model, C is the context and S_{test} is the test set.

Token Overlap: This is a relative metric which compares two output distributions over the vocabulary V. These distributions could be either produced by the same model on different inputs (in case of ICL: different number of demos, order of demos, etc.) or different models on the same inputs (ICL (with context) vs GD (fine-tuned, without context)). We sort the tokens based on their probability mass for each token and select the top-K tokens (denoted by T_K^1 and T_K^2). The token overlap is calculated as $\frac{1}{K}|T_K^1 \cap T_K^2|$. We use $K = 10$ in our experiments (most of the probability mass typically lies in top-10 tokens).

Overlap Cosine Similarity (OCS): Token overlap evaluates each of the top-K tokens with the same weight. With OCS, we measure how well the tokens agree individually. This metric is computed on the confidence distribution of top-

K tokens to avoid trivial values (most vocabulary tokens have low probabilities, making OCS ≈ 1). We denote the intersection of the two sets T_K^1, T_K^2 by $O = T_K^1 \cap T_K^2$ and use the following formula:

$$\text{OCS} = \frac{\sum_{t \in O} p^1(t) \cdot p^2(t)}{\sqrt{(\sum_{t \in O} p^1(t)^2) \cdot (\sum_{t \in O} p^2(t)^2) \cdot (K - |O|)}} \tag{6}$$

Intuitively, this quantifies the cosine distance between the overlapping tokens and assumes all the other tokens have zero overlap, therefore normalizing by $\sqrt{(K - |O|)}$ (when $K = |O|$, we divide by $\sqrt{1}$).

We evaluate every metric across three random seeds and compute the mean and std. Each random seed is used to sample demos for use in ICL experiments. The same demos are used to fine-tune models for GD/$\widehat{\text{GD}}$. Note that for *Token Overlap* and *Overlap Cosine Similarity*, the values for ICL are calculated between predictions made for the same set of demos but presented in a different order in the prompt.

5.2. Results

Gap between ICL and GD. Figure 5 shows our findings via plots of the three metrics, comparing ICL to various types of GD and /$\widehat{\text{GD}}$). We only show results for one dataset and one demonstration set size here. Other corresponding results are deferred to Appendix D and E due to space constraints. We see a clear gap between them ICL and all variants of GD and $\widehat{\text{GD}}$, across all three metrics, suggesting that these learning mechanisms likely work differently.

Comparing ICL vs. GD, ICL vs. ICL & GD vs. GD. In Figure 5, we see that the *Token Overlap* as well as *OCS* are consistently smaller between ICL and GD variants compared to ICL and ICL (with different demonstration order). For completeness, we conducted another experiment on AGNews where we calculated these relative metrics for different GD model checkpoints (say lr=1e-4 at epoch 20 and 1e-5 at epoch 200). Apart from the early epoch checkpoints (when most models have not changed much), most pairs had small *Token Overlap* and *OCS*. This shows how drastically GD based learning changes the model's behavior. With ICL–ICL comparisons, we see significantly higher values which point to a different functional behavior.

Why does GD perform poorly? As a trend in most datasets and setup variations, ICL outperforms GD and improves faster with increasing size of demonstration set (please see accuracy plots in Appendix D and E). This underlines our understanding about GD which tends to overfit when trained with only few samples. For illustration, we fine-tuned the model with GD using 512 demos and saw a boost in the performance (Table 1). Note that we can not compare this setting (with many demonstrations) with ICL because of the limited context window of LLaMa. Similar to our previous arguments, this also highlights that when a model performs ICL, it does not simply utilize demos like GD, but possibly recognizes the task from the demos and uses its prior knowledge about it to make predictions (Pan et al., 2023).

Table 1: Performance of GD (accuracy) increases with more samples, as expected. GD with many more demos obtains comparable performance to ICL with fewer demos, highlighting yet another empirical discrepancy.

Dataset	Demos	
	8	512
AGNews	0.42	0.69
CB	0.39	0.72
SST-2	0.49	0.75
RTE	0.36	0.65

Additional results on other datasets, with different numbers of ICL demos are deferred to Appendix D (GD) and Appendix E ($\widehat{\text{GD}}$). We also present other results about the impact of model size in Appendix F.

6. Related Work

Functional explanations. Many works offer functional explanations of ICL (Liu et al., 2022; Olsson et al., 2022; Schlag et al., 2021). Among these, explanations via GD (Garg et al., 2022; Zhang et al., 2023; Ahn et al., 2024) are most pertinent to our work. Notably, Akyürek et al. (2022) showed that Transformers can implement learning algorithms (gradient descent or closed-form OLS) for linear regression problems and empirically showed that the optimality of algorithms implemented experience a *phase shift* with increasing model size. Raventós et al. (2024) discovered similar results about algorithm discovery and phase shifts with increasing task diversity. Dai et al. (2023) similarly showed a dual between attention layers and linear layers optimized using gradient descent. Li et al. (2023) showed such an equivalence on softmax regression tasks. Finally, von Oswald et al. (2023) showed a similar construction with a simpler Linear Self-Attention Transformer, claiming that Transformers learn in-context using gradient descent on linear regression problems. Notably, Akyürek et al. (2022) found this GD behavior applicable only in small models, with bigger models exhibiting Bayes optimal learning behavior (like Ordinary Least Squares for linear regression). In contrast, von Oswald et al. (2023) claimed that bigger Transformers also implement GD with added data transformations.

Most of this line of work shows how Transformers have the ability to implement such algorithms resulting from training on ICL objectives (**hypothesis 2**) and not that real-world models pretrained on natural data develop this ability (**hypothesis 1**).

Distributional explanations. This body of work explains ICL via distributional frameworks and the relevant properties of LLMs (Xie et al., 2021; Wies et al., 2024). Xie et al. (2021) explained ICL as implicit Bayesian inference, which implicitly maps a given set of demonstrations to an appropriate latent concept (task) learned via pretraining on a massive unsupervised corpus. Similarly, Hahn & Goyal (2023) theorized that natural language pretraining data consists of compositional structure, which leads to the emergent ability of in-context learning, while Chan et al. (2022) showed that this might be because of distributional properties of the training distribution (like burstiness). These are all reasonable explanations of how ICL works, although they are somewhat tangential to the focus of this study.

Empirical studies. Various empirical works study ICL under various settings (Brown et al., 2020; Zhao et al., 2021; Min et al., 2022; Mishra et al., 2022; Han et al., 2023; Wang et al., 2023). To note a few, Srivastava et al. (2023) famously benchmarked ICL for many tasks and models. Perez et al. (2021); Lu et al. (2022) showed the sensitivity of ICL to the choice of demonstrations and their orderings. Shin et al. (2022); Razeghi et al. (2022) showed the sensitivity of ICL performance to the frequency and size of the relevant pretraining corpus. Shen et al. (2023) treat the ICL prompt selection as an optimization problem. Pan et al. (2023) disentangle task recognition and task learning in ICL, which is analyzed in theory recently by Lin & Lee (2024). These works highlight numerous ways the ability of models to perform ICL changes under different conditions but do not attempt to explain how it functions.

7. Discussion and Conclusion

This work intends to clarify the distinction between naturally emergent ICL (commonly seen in LLMs pretrained on natural text data); hypothesis 1) vs. task-specific ICL as a result of training Transformers for ICL (hypothesis 2). While recent work has shown that Transformers have the ***expressive capacity*** to simulate gradient-descent in their forward pass, this does ***not*** immediately imply that real-world models ***actually do*** simulate it. We hope this work motivates alternative approaches that reveal the true nature of in-context learning in pretrained LLMs.

We recognize that hypothesis 1 establishing a universal equivalence between ICL and GD may be too strong. A more reasonable hypothesis might involve certain restrictions, such as the target task's distributional properties or the number of demonstrations. However, the specifics of such conditions are unclear, so we have opted for a general statement.

Besides using in-context demonstrations, recent work has also discovered other ways in which in-context prompts enhance the performance of LLMs. For example, appending prompts like "*Think step by step*" (Kojima et al., 2022) or "*Take a deep breath and think*" (Yang et al., 2023) before asking a task-specific question has been shown to improve zero-shot performance of LLMs. Such evidence may suggest that an optimization algorithm like GD cannot fully describe the ability of ICL. Understanding ICL dynamics requires a more holistic theory which considers the various nuances of this remarkable learning paradigm.

8. Limitations and Future Opportunities

Because of its computationally infeasible nature, we were not able to do an exhaustive search over all sub-models and pinpoint which subset of parameters could correspond to sub-models that could get updated in $\widehat{\text{GD}}$. This could be an interesting avenue of research. Moreover, we do not provide alternate explanations of how ICL works functionally. As ICL is hard to study directly in LLMs, it is natural to turn to simpler settings. But it is imperative that we keep the setups analogous so that inferences from one can be extended to the other.

Impact Statement

It is evident that LLMs and their remarkable ability to learn in context have far-reaching impacts in various applications. Understanding the nuances of ICL and its exact functional behavior will uncover the true strengths and limits of LLMs, which is essential to use them reliably. A growing line of research shows theoretical expressivity of transformers to simulate gradient descent by training them on ICL objectives. But it is important to differentiate this from the natural ICL that emerges in language models, so that progress towards understanding its true nature is made in the right direction.

Acknowledgements

This work is supported in part by ONR grant N00014-24-1-2089, and generous gifts from Amazon and the Allen Institute for AI. We are grateful to the anonymous reviewers for constructive feedback for improving this work. We also thank Anqi Liu, Jason Eisner, Holden Lee, Tianjian Li and the anonymous reviewers for their insightful discussions. GPU machines for conducting experiments were provided by ARCH Rockfish cluster at Johns Hopkins University (`https://www.arch.jhu.edu`).

References

Ahn, K., Cheng, X., Daneshmand, H., and Sra, S. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Advances in Neural Information Processing Systems* (NeurIPS), 2024. URL https://arxiv.org/abs/2306.00297.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models. In *International Conference on Learning Representations* (ICLR), 2022. URL https://arxiv.org/abs/2211.15661.

Black, S., Gao, L., Wang, P., Leahy, C., and Biderman, S. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. March 2021. doi: 10.5281/zenodo.5297715. URL https://doi.org/10.5281/zenodo.5297715.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems* (NeurIPS), 2020. URL https://arxiv.org/abs/2005.14165.

Chan, S., Santoro, A., Lampinen, A., Wang, J., Singh, A., Richemond, P., McClelland, J., and Hill, F. Data distributional properties drive emergent in-context learning in transformers. *Advances in Neural Information Processing Systems* (NeurIPS), 35:18878–18891, 2022. URL https://arxiv.org/abs/2205.05055.

Chiang, D., Cholak, P., and Pillay, A. Tighter bounds on the expressivity of transformer encoders. In *International Conference on Machine Learning* (ICML), 2023. URL https://arxiv.org/abs/2301.10743.

Dagan, I., Glickman, O., and Magnini, B. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, 2005. URL https://link.springer.com/chapter/10.1007/11736790_9.

Dai, D., Sun, Y., Dong, L., Hao, Y., Sui, Z., and Wei, F. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. In *Annual Meeting of the Association for Computational Linguistics* (ACL) - *Findings*, 2023. URL https://arxiv.org/abs/2212.10559.

De Marneffe, M.-C., Simons, M., and Tonhauser, J. The CommitmentBank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, 2019. URL https://ojs.ub.uni-konstanz.de/sub/index.php/sub/article/view/601.

Garg, S., Tsipras, D., Liang, P. S., and Valiant, G. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems* (NeurIPS), 35:30583–30598, 2022. URL https://arxiv.org/abs/2208.01066.

Hahn, M. and Goyal, N. A theory of emergent in-context learning as implicit structure induction. *arXiv preprint arXiv:2303.07971*, 2023. URL https://arxiv.org/abs/2303.07971.

Han, X., Simig, D., Mihaylov, T., Tsvetkov, Y., Celikyilmaz, A., and Wang, T. Understanding in-context learning via supportive pretraining data. In *Annual Meeting of the Association for Computational Linguistics* (ACL), 2023. URL https://arxiv.org/abs/2306.15091.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems* (NeurIPS), 2022. URL https://arxiv.org/abs/2205.11916.

Li, S., Song, Z., Xia, Y., Yu, T., and Zhou, T. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*, 2023. URL https://arxiv.org/abs/2304.13276.

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

Liu, B., Ash, J. T., Goel, S., Krishnamurthy, A., and Zhang, C. Transformers learn shortcuts to automata. In *International Conference on Learning Representations* (ICLR), 2022. URL https://arxiv.org/abs/2210.10749.

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., and Shazeer, N. Generating wikipedia by summarizing long sequences. In *International Conference on Learning Representations* (ICLR), 2018. URL https://arxiv.org/abs/1801.10198.

Lu, Y., Bartolo, M., Moore, A., Riedel, S., and Stenetorp, P. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Annual Meeting of the Association for Computational Linguistics* (ACL), 2022. URL https://arxiv.org/pdf/2104.08786.pdf.

Merrill, W., Sabharwal, A., and Smith, N. A. Saturated transformers are constant-depth threshold circuits. *Transactions of the Association for Computational Linguistics* (TACL), 10:843–856, 2022. URL https://arxiv.org/abs/2106.16213.

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work? In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2022. URL https://arxiv.org/abs/2202.12837.

Mishra, S., Khashabi, D., Baral, C., Choi, Y., and Hajishirzi, H. Reframing instructional prompts to gptk's language. In *Annual Meeting of the Association for Computational Linguistics* (ACL) - *Findings*, 2022. URL https://arxiv.org/abs/2109.07830.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022. URL https://arxiv.org/abs/2209.11895.

Pan, J., Gao, T., Chen, H., and Chen, D. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, July 2023. URL https://aclanthology.org/2023.findings-acl.527.

Perez, E., Kiela, D., and Cho, K. True few-shot learning with language models. In *Advances in Neural Information Processing Systems* (NeurIPS), 2021. URL https://proceedings.neurips.cc/paper/2021/file/5c04925674920eb58467fb52ce4ef728-Paper.pdf.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019. URL https://openai.com/blog/better-language-models/.

Raventós, A., Paul, M., Chen, F., and Ganguli, S. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. In *Advances in Neural Information Processing Systems* (NeurIPS), 2024.

Razeghi, Y., Logan IV, R. L., Gardner, M., and Singh, S. Impact of pretraining term frequencies on few-shot reasoning. In *Annual Meeting of the Association for Computational Linguistics* (ACL) - *Findings*, 2022. URL https://arxiv.org/abs/2202.07206.

Schlag, I., Irie, K., and Schmidhuber, J. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning* (ICML), pp. 9355–9366, 2021. URL http://proceedings.mlr.press/v139/schlag21a.html.

Shen, L., Tan, W., Zheng, B., and Khashabi, D. Flatness-aware prompt selection improves accuracy and sample efficiency. *arXiv preprint arXiv:2305.10713*, 2023. URL https://arxiv.org/abs/2305.10713.

Shin, S., Lee, S. W., Ahn, H., Kim, S., Kim, H. S., Kim, B., Cho, K., Lee, G., Park, W., Ha, J. W., et al. On the effect of pretraining corpora on in-context learning by a large-scale language model. In *Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL), 2022. URL https://arxiv.org/abs/2204.13509.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), pp. 1631–1642, 2013. URL https://aclanthology.org/D13-1170.pdf.

Srivastava, A., Rastogi, A., Rao, A., Shoeb, A. A. M., Abid, A., Fisch, A., Brown, A. R., Santoro, A., Gupta, A., Garriga-Alonso, A., et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research* (TMLR), 2023. URL https://arxiv.org/abs/2206.04615.

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

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is All You Need. In *Advances in Neural Information Processing Systems* (NeurIPS), 2017. URL https://arxiv.org/abs/1706.03762.

von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. In *International Conference on Learning Representations* (ICLR), pp. 35151–35174, 2023. URL https://arxiv.org/abs/2212.07677.

Wang, B. and Komatsuzaki, A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. Self-Instruct: Aligning Language Model with Self Generated Instructions. In

Annual Meeting of the Association for Computational Linguistics (ACL), 2023. URL https://arxiv.org/abs/2212.10560.

Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations* (ICLR), 2021. URL https://arxiv.org/abs/2109.01652.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022. URL https://arxiv.org/abs/2206.07682.

Wies, N., Levine, Y., and Shashua, A. The learnability of in-context learning. In *Advances in Neural Information Processing Systems* (NeurIPS), 2024. URL https://arxiv.org/abs/2303.07895.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2111.02080.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers. In *International Conference on Learning Representations* (ICLR), 2023. URL https://arxiv.org/abs/2309.03409.

Zhang, R., Frei, S., and Bartlett, P. L. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*, 2023. URL https://arxiv.org/abs/2306.09927.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems* (NeurIPS), 2015. URL https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

Zhao, Z., Wallace, E., Feng, S., Klein, D., and Singh, S. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning* (ICML), pp. 12697–12706, 2021. URL http://proceedings.mlr.press/v139/zhao21c/zhao21c.pdf.

Supplementary Material

A. Order sensitivity of ICL and GD-based algorithms

We present empirical evidence highlighting the distinct sensitivities of GD-based algorithms and ICL with respect to data order. Specifically, we assess the variation in confidence assigned to vocabulary V by the model across different data orderings.

Experimental setup We evaluate the order sensitivity of GD-based algorithms using the GD, SGD, and Adam optimizers. The chosen learning rates are 1e-4, 1e-5, 5e-4, and 5e-5. Our experiments are conducted on the AGNews dataset using the LLaMa-7B model. We set the number of demonstrations to 8. GD training continues for 200 epochs to avoid issues of non-convergence, but is evaluated at every 20 epochs. The number N of random orders $\{\sigma_i\}_{i=1}^{N}$ is set as 10 (as the total number of orders are combinatorial).

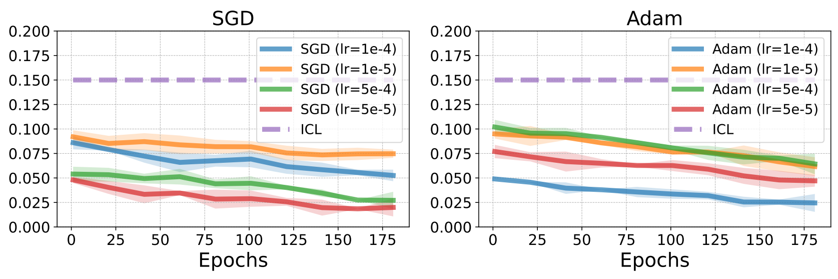
Evaluation metric (Sen) As for the evaluation metric of sensitivity (Sen), it is defined as follows: Given a set of confidence vectors $\{p_i\}_{i=1}^{N}$ resulting from distinct data orders $\{\sigma_i\}_{i=1}^{N}$, we calculate the standard deviation for each dimensionality within V using the samples $\{p_i\}_{i=1}^{N}$. Subsequently, the variances for individual tokens are aggregated.

Results In Figure 4, we presented a high level overview of our findings. In Figure 6, we present it in detail. First, ICL exhibits a much more pronounced data order sensitivity than the three GD-based algorithms. Second, as GD training progresses, its sensitivity diminishes. And third, this happens with both GD and $\widehat{\text{GD}}$. Overall, these findings underscore distinct behaviors of ICL and GD-based algorithms with respect to data order. This suggests a disparity between ICL and GD, as shown in Theorem 1.

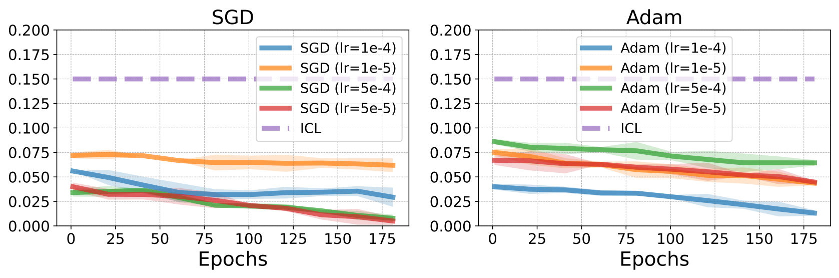
Ablation results

Batch size: In Figure 7, we show that a similar trend is seen when we ablate the batch size.

Model: This difference in order sensitivity is not restricted to the LLaMa model. In Figure 8, we show an experiment with the AGNews dataset, where the order sensitivity of ICL is similarly higher than GD variants for other LLMs (like Qwen-7B and GPT-J).

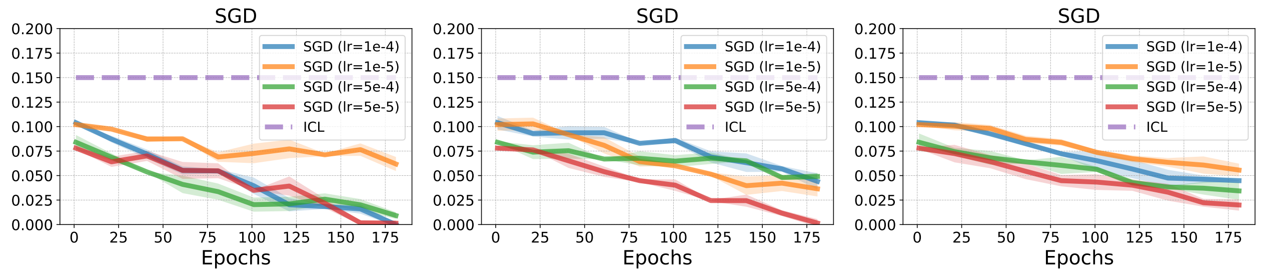


(a) Order sensitivity of ICL and GD when batchsize = 1

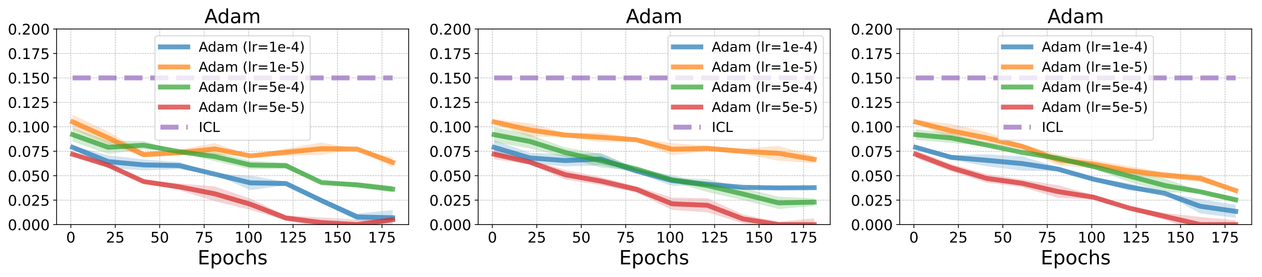


(b) Order sensitivity of ICL and $\widehat{\text{GD}}$ when batchsize = 4

Figure 6: The order sensitivity (y-axis represents Sen (appendix A)) of ICL and GD (SGD and Adam) as the batchsize changes.

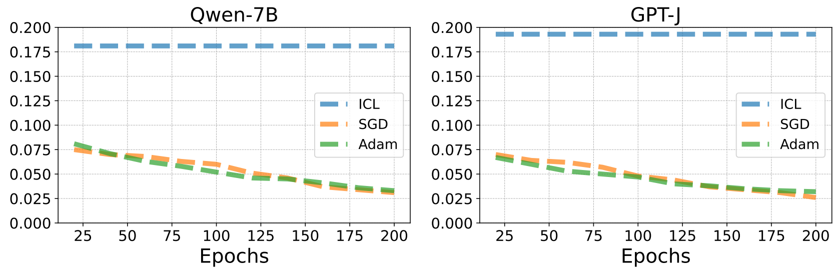


(a) Order sensitivity of ICL and $\widehat{\text{GD}}$ (SGD)



(b) Order sensitivity of ICL and $\widehat{\text{GD}}$ (Adam)

Figure 7: The order sensitivity (y-axis represents Sen (appendix A)) of ICL and $\widehat{\text{GD}}$ (SGD and Adam) as the batchsize changes. From left to right, three figures refer to cases bs=1, 2, 4.



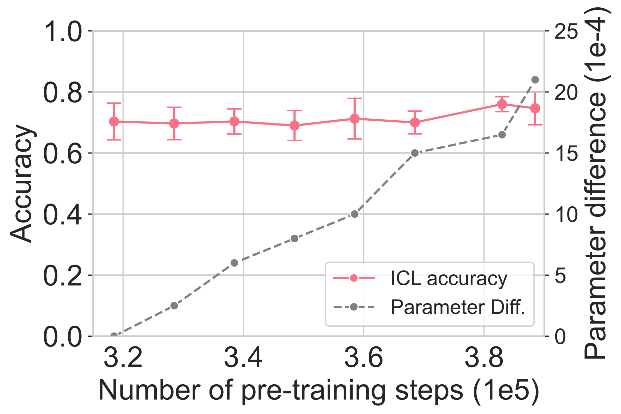
(a) Order sensitivity of ICL and GD

Figure 8: The order sensitivity (y-axis represents Sen (appendix A)) of ICL and GD (SGD and Adam) on Qwen-7B and GPT-J. The dataset is AGNews, and the batchsize is set as 4.

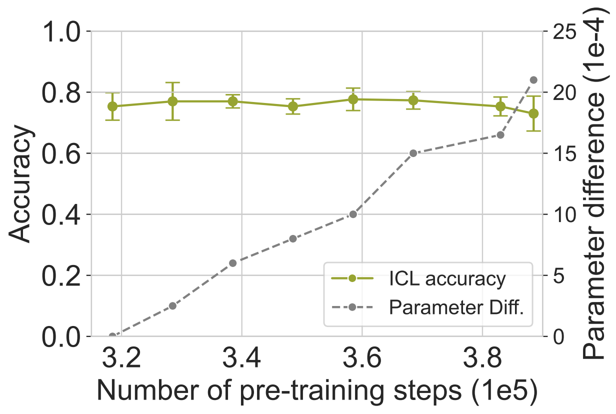
B. How does ICL evolve during training?

Experimental setup. We chose intermediate checkpoints from GPT-J, ranging from 310k to 380k pretraining steps. Using these varied pretraining steps, our approach simulates the fine-tuning process. Specifically, we focus on two metrics to quantify the magnitude of fine-tuning: (1) Step Gap: This represents the difference in pretraining steps between selected checkpoints. (2) Parameter Gap: In line with the assumptions made by Oswald et al. (von Oswald et al., 2023), we compute the average differences for each parameter within the W_K, W_Q, and W_V matrices across different checkpoints. To evaluate the ICL capacity of the models, we conducted tests on AGNews, SST-2, CB, and RTE using eight demonstrations.

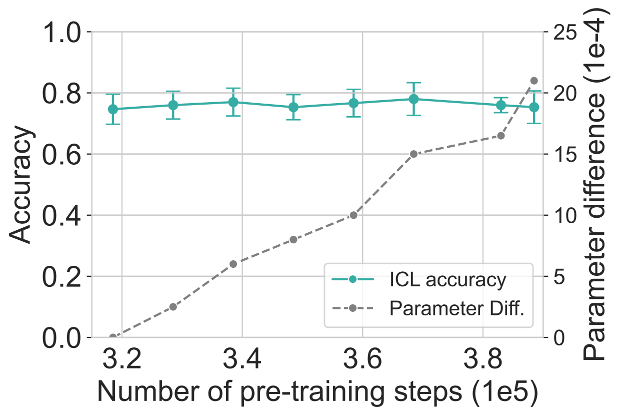
Results. The results are shown in Figure 9, from where we can observe that there is no significant gap between ICL capacity of different checkpoints, indicating that continued fine-tuning (pretraining) will not substantially hurt the ICL performance.



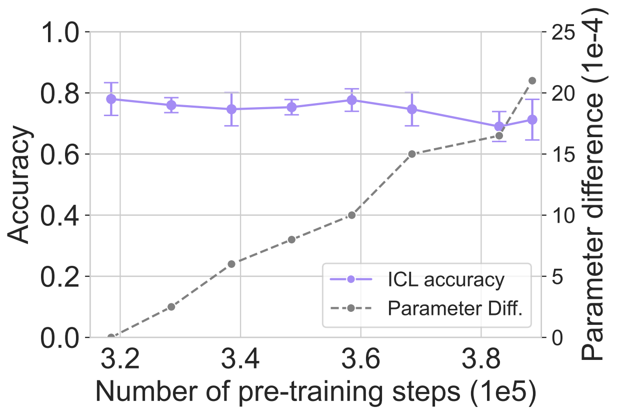
(a) AGNews



(b) SST-2



(c) CB

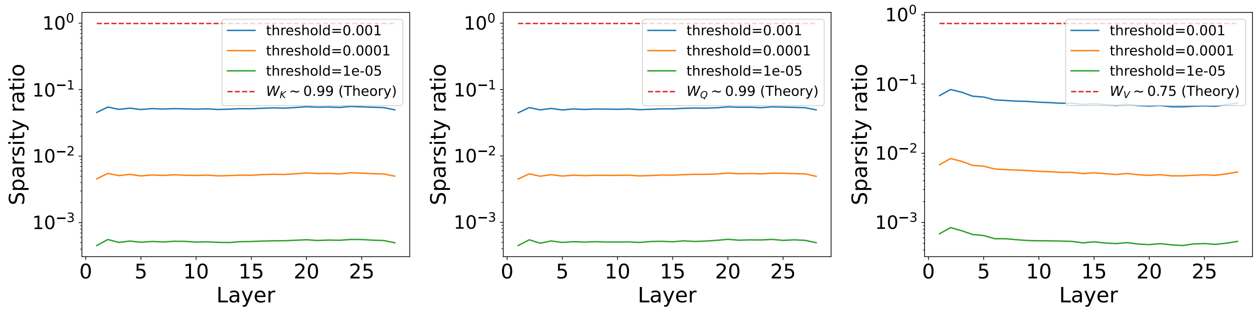


(d) RTE

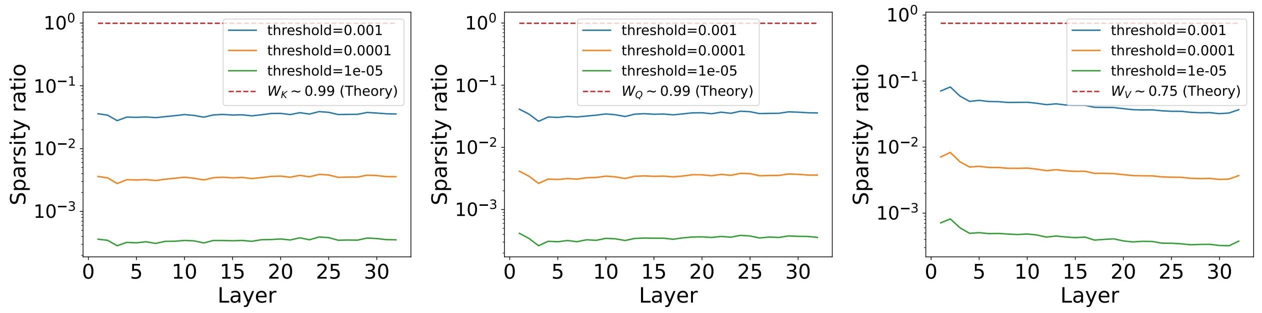
Figure 9: The ability of GPT-J to perform ICL does not change much over a time cross-section of training while the parameters change steadily.

C. Layer-wise sparsity rate of LLMs

We show the sparsity ratio of each layer of LLMs. Specifically, in our paper, we have used LLaMa-7B and GPT-J are main experiments, so we show their sparsity rate of W_K, W_Q, and W_V in each layer. The results are shown in Figure 10. It is interesting that although W_K and W_Q have almost constant sparsity in all layers, W_V has slightly decaying sparsity.



(a) Sparsity ratio of LLaMa-7B



(b) Sparsity ratio of GPT-J

Figure 10: The sparse ratio of LLaMa-7B and GPT-J in each layer. From left to right, three figures represent the cases of W_K, W_Q, and W_V.

D. Additional results on ICL vs GD comparisons

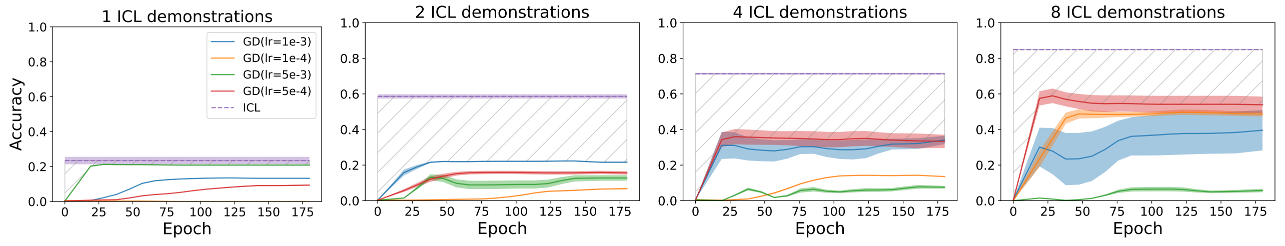
Here we present all comprehensive plots on ICL vs GD on AGNews and other datasets.

The case of $N = 1$. We see an almost similar accuracy between ICL and one GD variant in all datasets, which is an interesting finding. There are several reasons why this does not directly imply ICL≈GD:

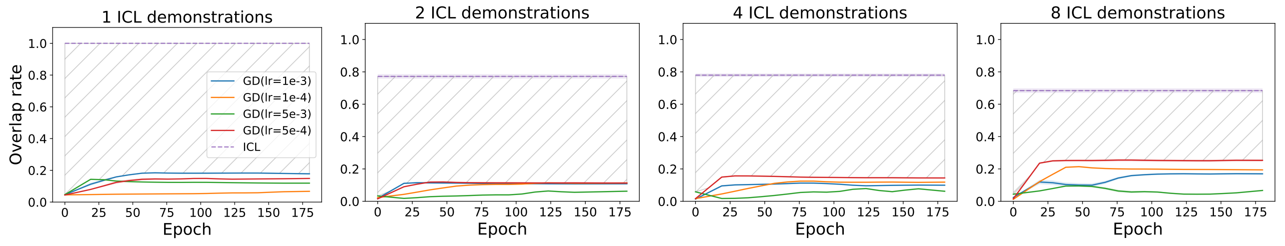
1. There are different GD variants that correspond to the ICL performance in each dataset. This implies the absence of a standard GD-like algorithm that would work on all problems.

2. Other nuanced metrics show that there is a stark difference in the output distributions of ICL and all GD variants.

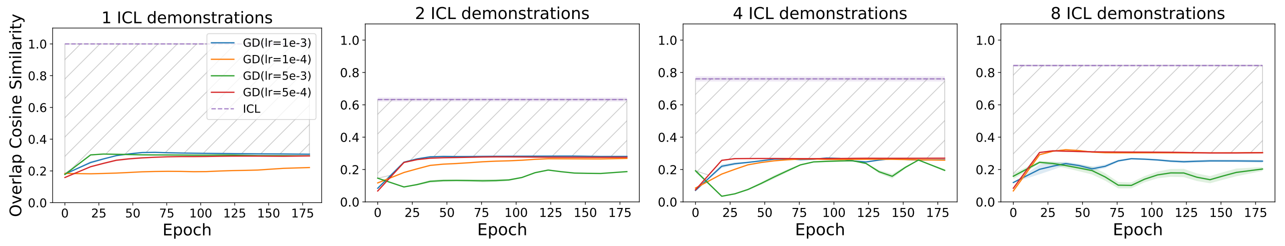
3. The jump in performance from $N = 1$ to $N = 2$ is typically much more pronounced for ICL than GD. This hints at differences in their functional behavior.



(a) *Accuracy* comparison

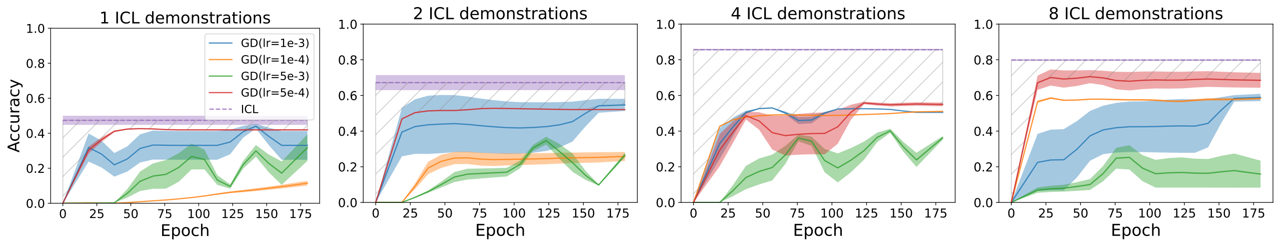


(b) *Token overlap* comparison

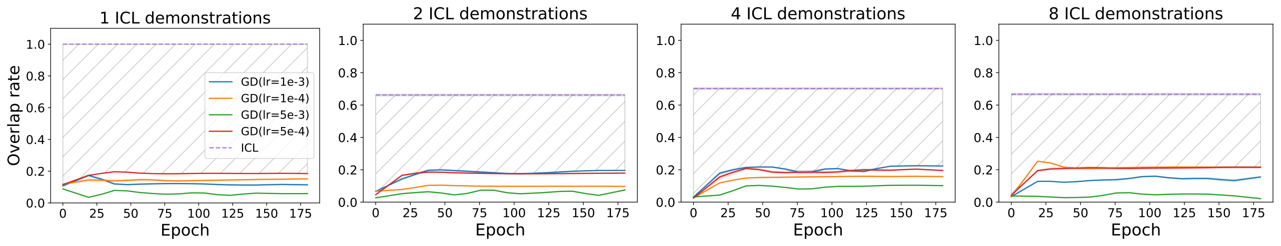


(c) *Overlap Cosine Similarity* comparison

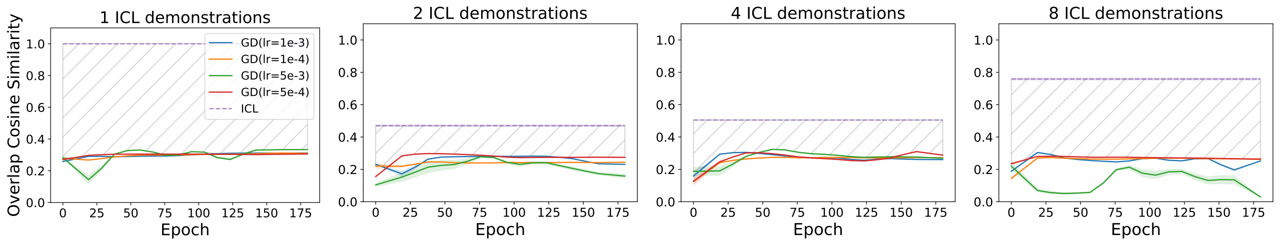
Figure 11: Comparison of ICL and GD for the AGNews dataset, with increasing number of demonstrations.



(a) *Accuracy* comparison

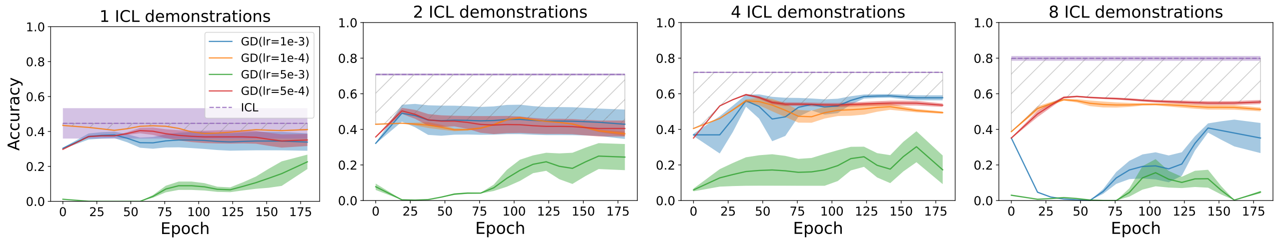


(b) *Token overlap* comparison

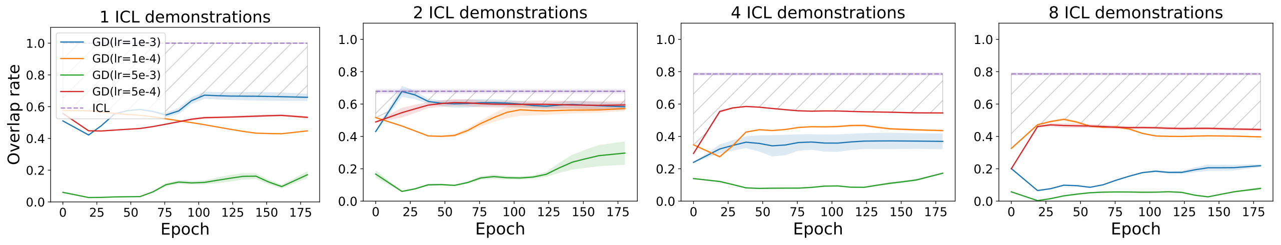


(c) *Overlap Cosine Similarity* comparison

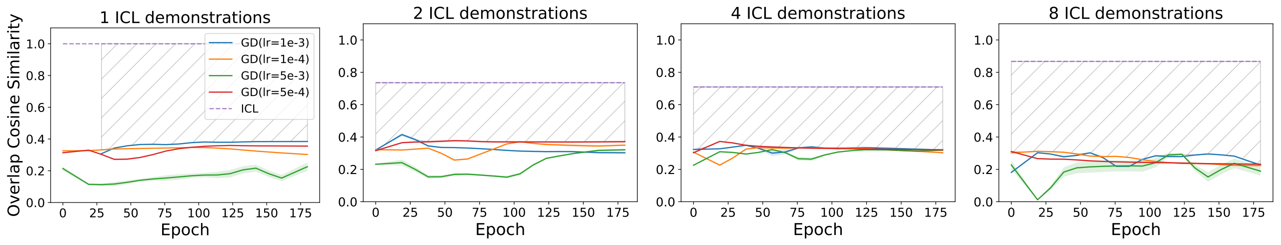
Figure 12: Comparison of ICL and GD for the SST dataset, with increasing number of demonstrations.



(a) *Accuracy* comparison

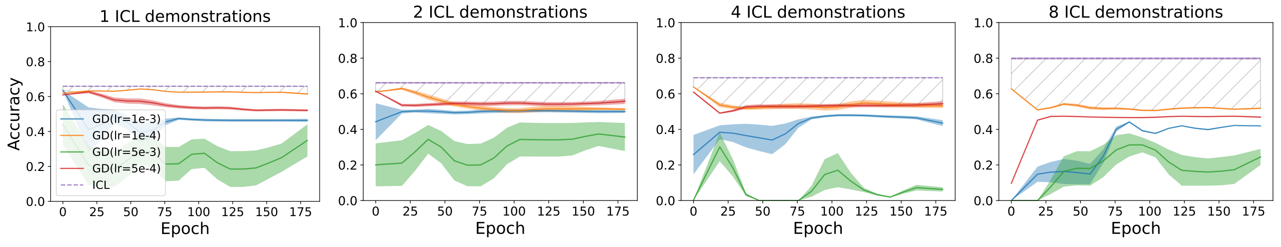


(b) *Token overlap* comparison

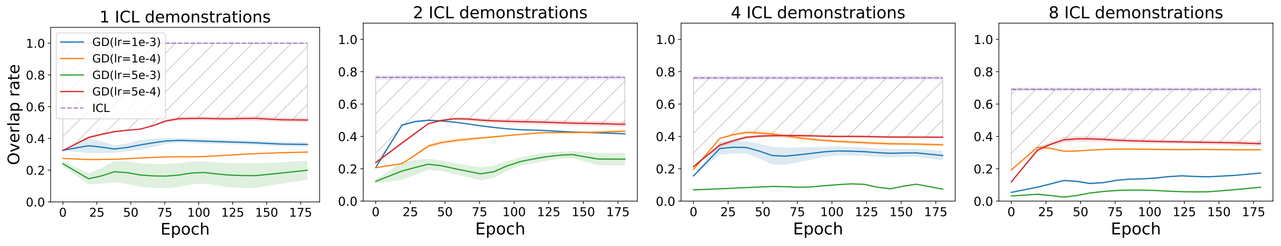


(c) *Overlap Cosine Similarity* comparison

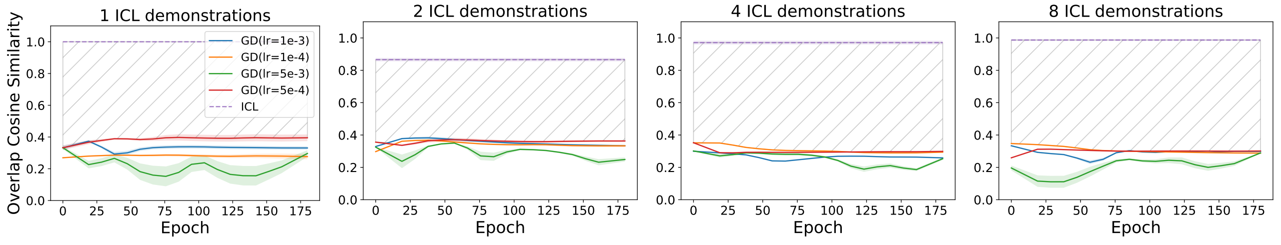
Figure 13: Comparison of ICL and GD for the CB dataset, with increasing number of demonstrations.



(a) *Accuracy* comparison



(b) *Token overlap* comparison



(c) *Overlap Cosine Similarity* comparison

Figure 14: Comparison of ICL and GD for the RTE dataset, with increasing number of demonstrations.

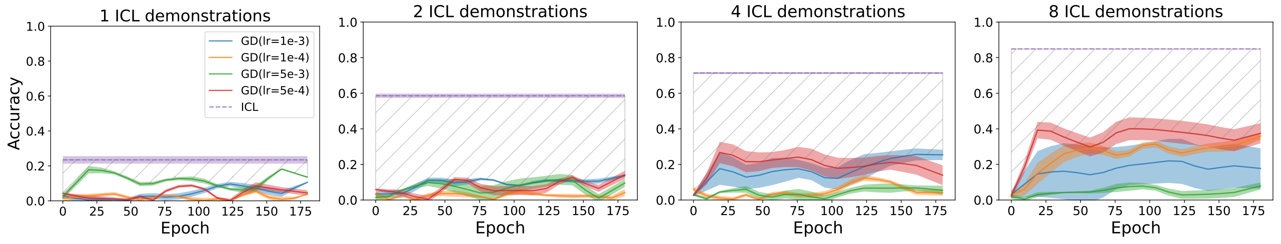
E. Empirical results on ICL vs $\widehat{\mathrm{GD}}$

Here, we present corresponding results on ICL vs $\widehat{\mathrm{GD}}$.

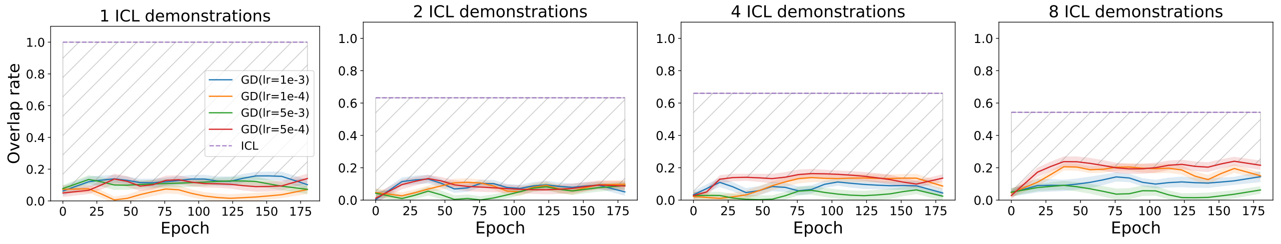
How are sub-models selected for optimization? Since $\widehat{\mathrm{GD}}$ conducts updates only on the subset of the model and enumerating all the possible subsets of model parameters is infeasible, we select intuitive subsets of parameters to simulate $\widehat{\mathrm{GD}}$.

We use the hypotheses in (Akyürek et al., 2022; von Oswald et al., 2023), to experiment with intuitive subsets of models. In particular, according to von Oswald et al. (2023) the *implicit* model lies in W_V of the Transformer while the probing experiments in (Akyürek et al., 2022) suggest that this iterative optimization happens in top layers of the Transformers. Therefore, we provide experiments with two intuitive subsets to simulate $\widehat{\mathrm{GD}}$: finetuning (1) W_V of a single deep layer, and (2) W_V of a single middle layer.

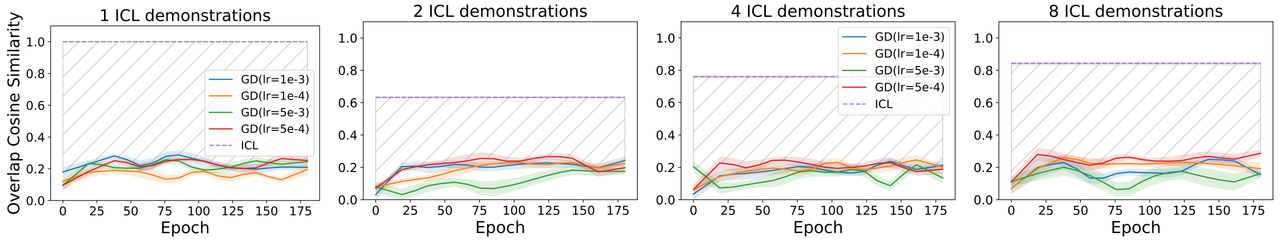
Results of ICL vs. $\widehat{\mathrm{GD}}$ (Deep layer) Following a similar experimental setup in §5, we compare the differences between ICL and $\widehat{\mathrm{GD}}$. We randomly select one layer from the last four layers from LLaMa (29-32), repeat the experiments four times and plot the mean and std. The results are shown in Figure 15 - Figure 18, and we can observe similar gaps between ICL and $\widehat{\mathrm{GD}}$.



(a) *Accuracy* comparison

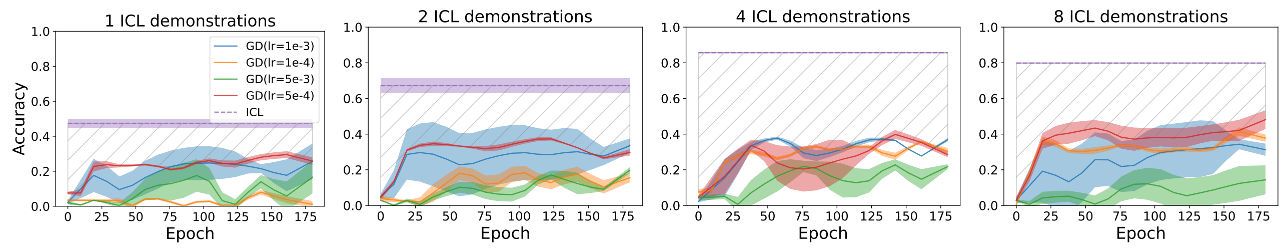


(b) *Token overlap* comparison

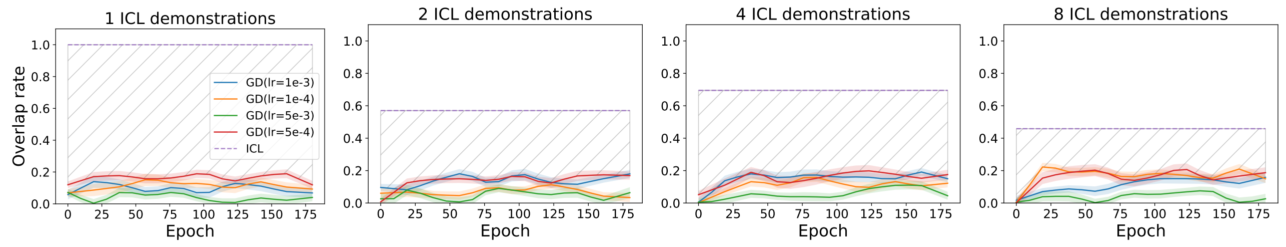


(c) *Overlap Cosine Similarity* comparison

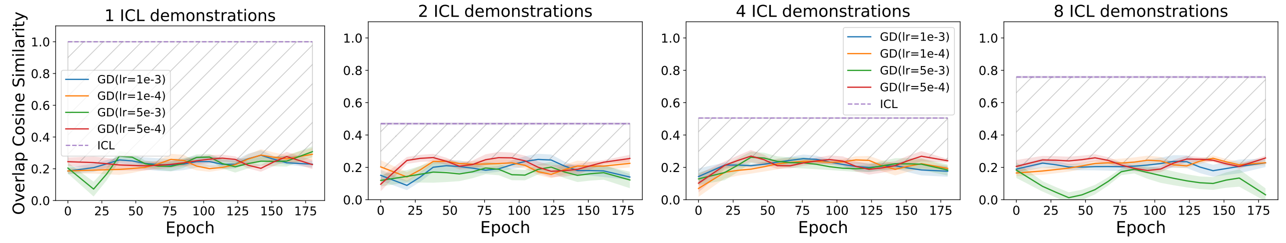
Figure 15: Comparison of ICL and $\widehat{\mathrm{GD}}$ for the AGNews dataset, with increasing number of demonstrations. $\widehat{\mathrm{GD}}$ is simulated by optimizing on one random deep layer of LLaMa.



(a) *Accuracy* comparison

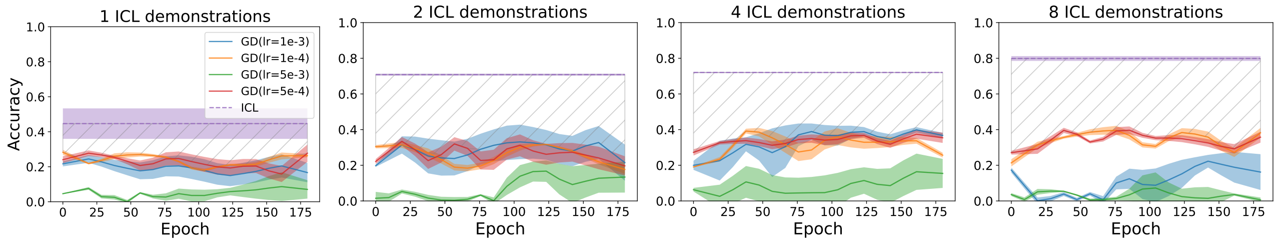


(b) *Token overlap* comparison

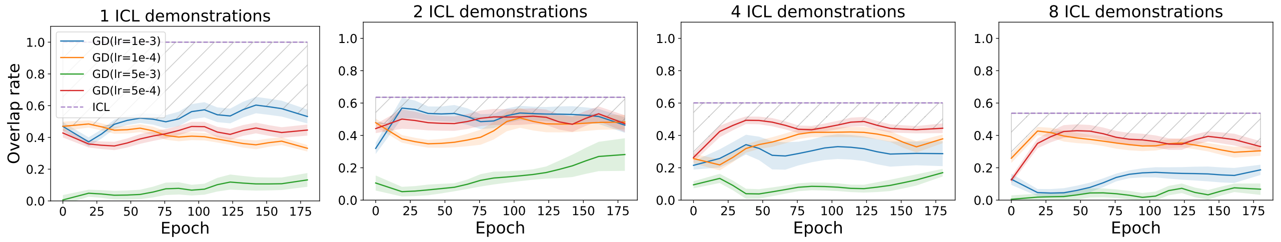


(c) *Overlap Cosine Similarity* comparison

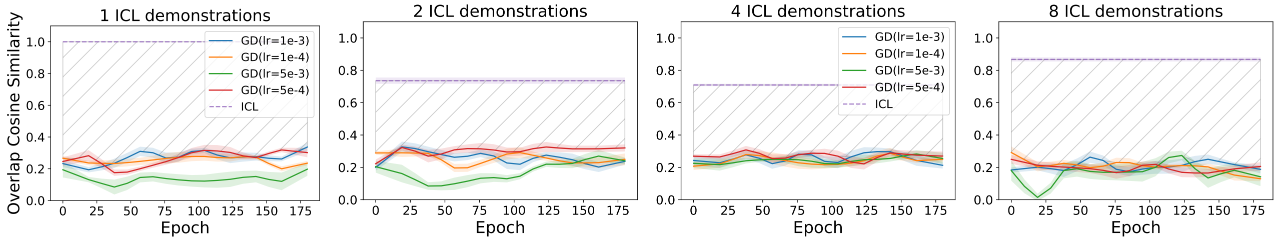
Figure 16: Comparison of ICL and $\widehat{\text{GD}}$ for the SST dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random deep layer of LLaMa.



(a) *Accuracy* comparison

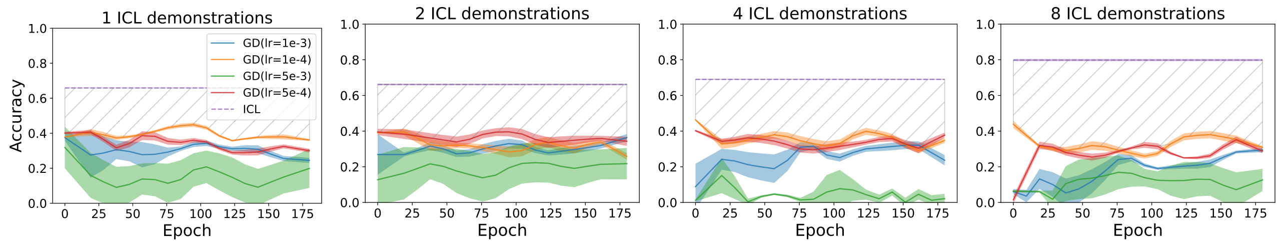


(b) *Token overlap* comparison

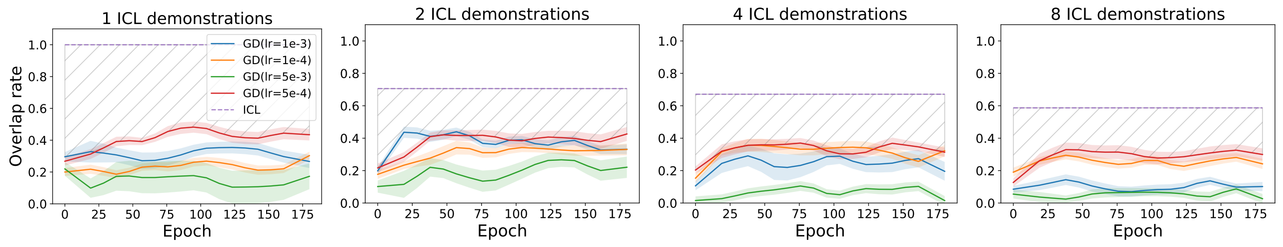


(c) *Overlap Cosine Similarity* comparison

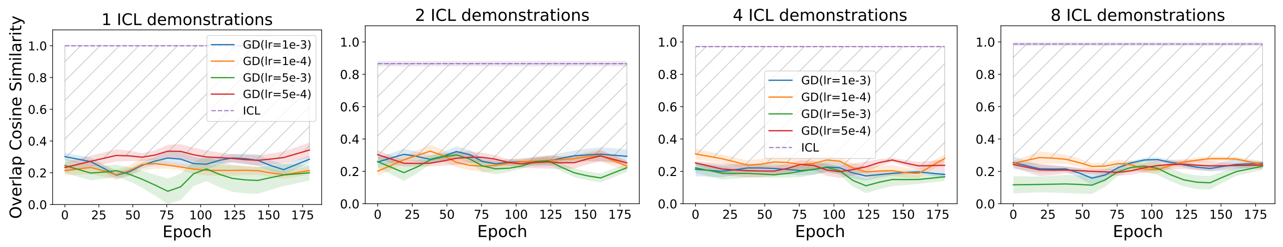
Figure 17: Comparison of ICL and $\widehat{\text{GD}}$ for the CB dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random deep layer of LLaMa.



(a) *Accuracy* comparison



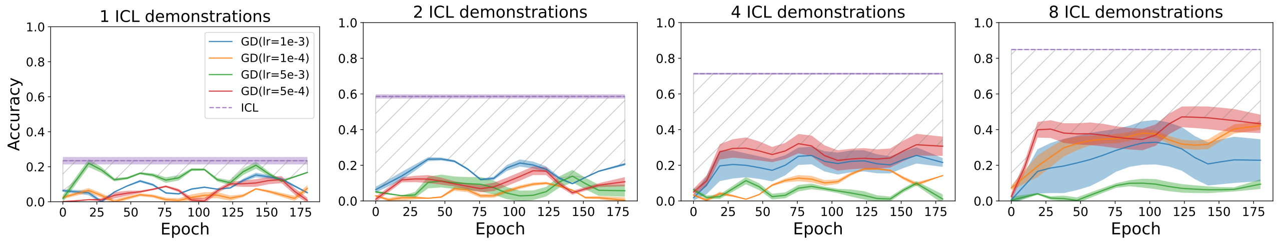
(b) *Token overlap* comparison



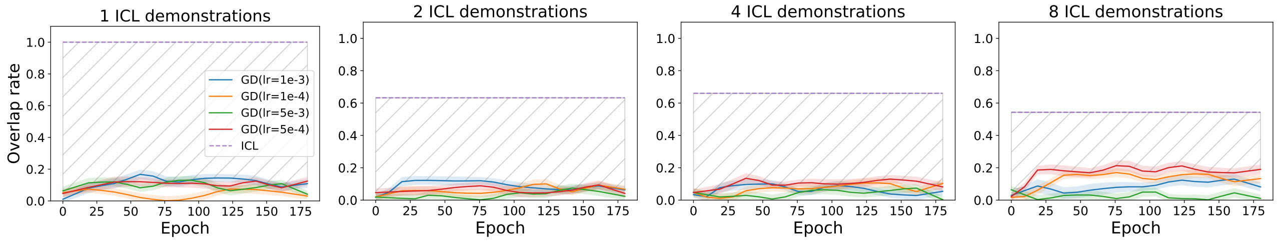
(c) *Overlap Cosine Similarity* comparison

Figure 18: Comparison of ICL and $\widehat{\text{GD}}$ for the RTE dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random deep layer of LLaMa.

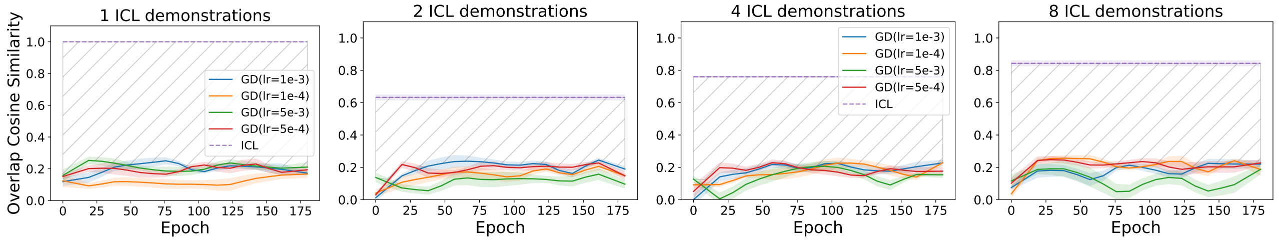
Results of ICL vs. $\widehat{\textbf{GD}}$ (Middle layers) This time, we randomly select one layer from the middle layers of LLaMa (16-20). The results are shown in Figure 19 - Figure 22, we can observe similar gaps between ICL and $\widehat{\text{GD}}$.



(a) *Accuracy* comparison

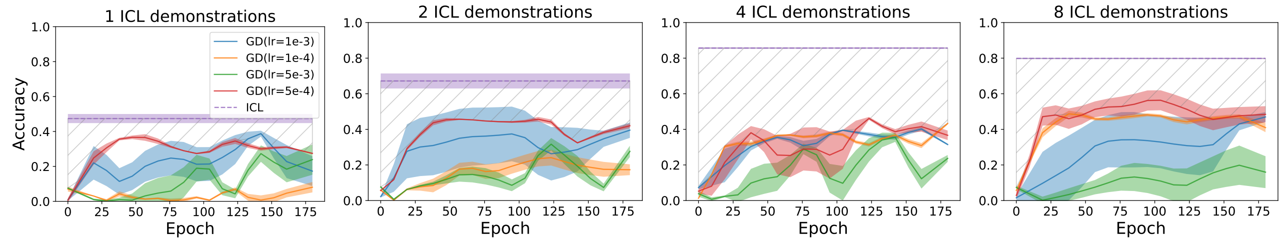


(b) *Token overlap* comparison

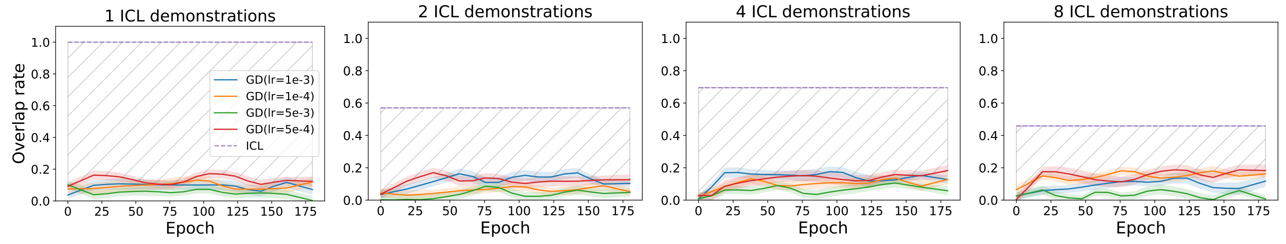


(c) *Overlap Cosine Similarity* comparison

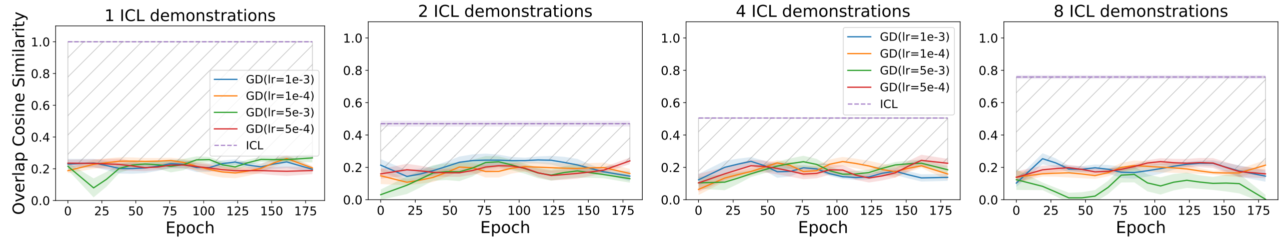
Figure 19: Comparison of ICL and $\widehat{\text{GD}}$ for the AGNews dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random middle layer of LLaMa.



(a) *Accuracy* comparison

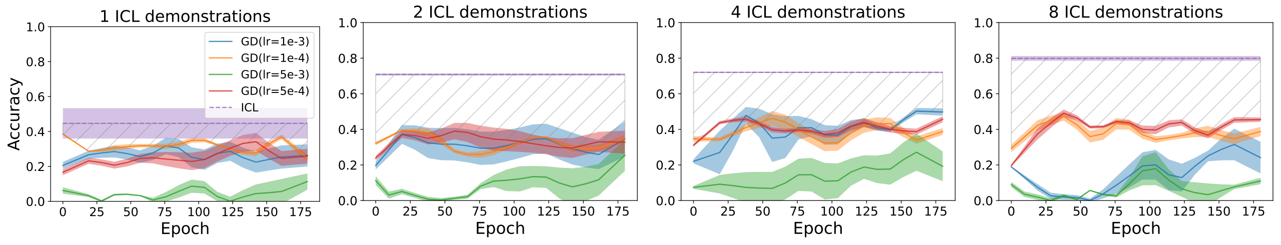


(b) *Token overlap* comparison

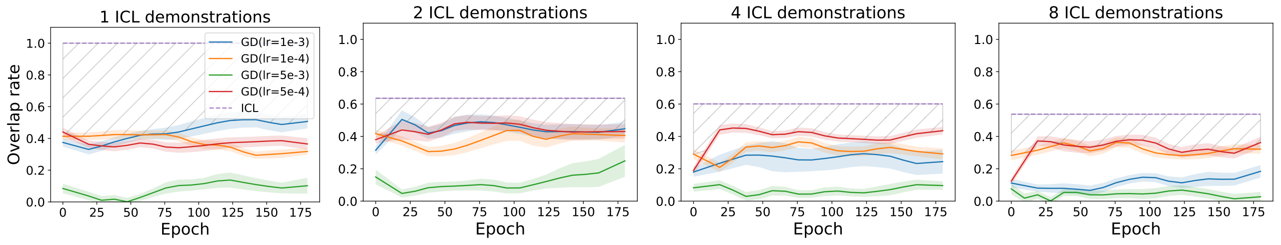


(c) *Overlap Cosine Similarity* comparison

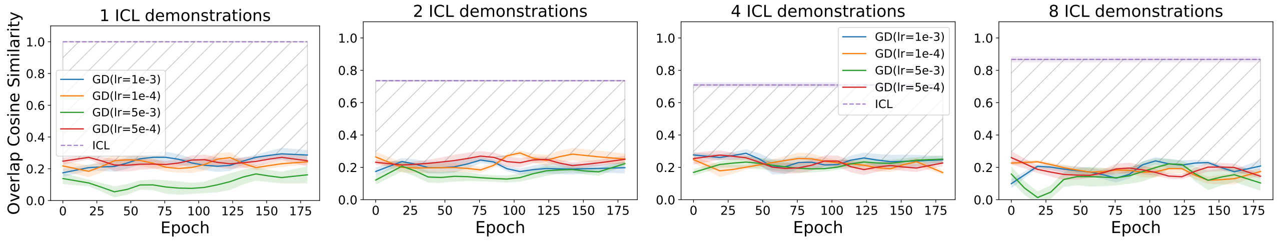
Figure 20: Comparison of ICL and $\widehat{\text{GD}}$ for the SST dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random middle layer of LLaMa.



(a) *Accuracy* comparison

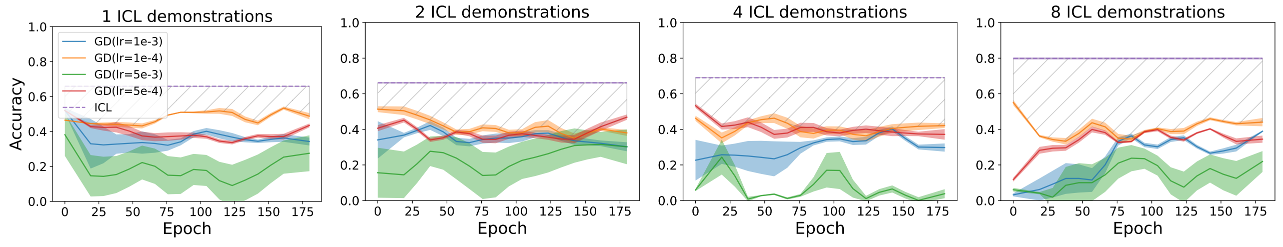


(b) *Token overlap* comparison

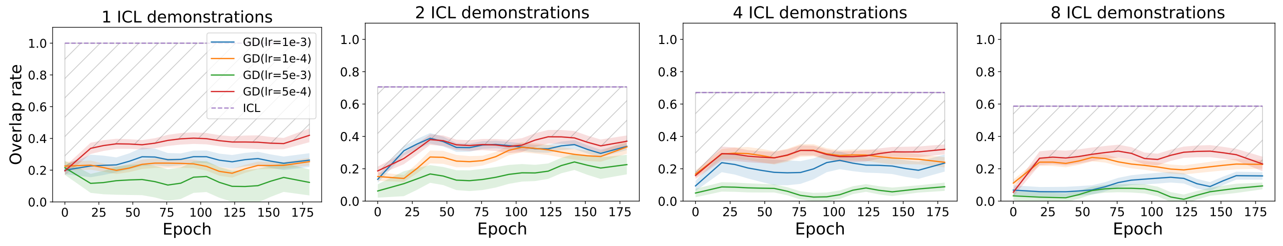


(c) *Overlap Cosine Similarity* comparison

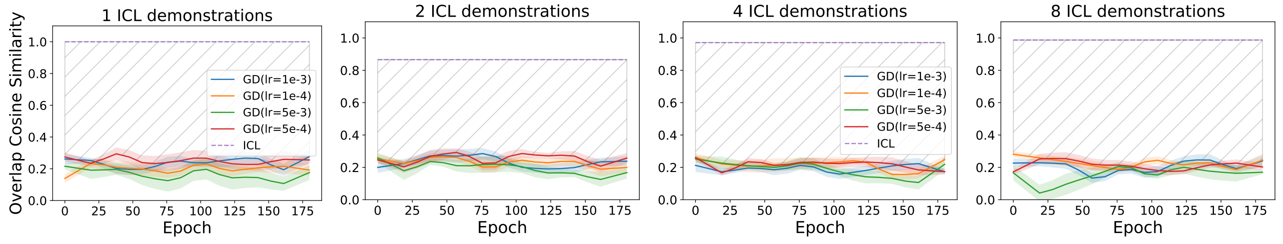
Figure 21: Comparison of ICL and $\widehat{\text{GD}}$ for the CB dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random middle layer of LLaMa.



(a) *Accuracy* comparison



(b) *Token overlap* comparison

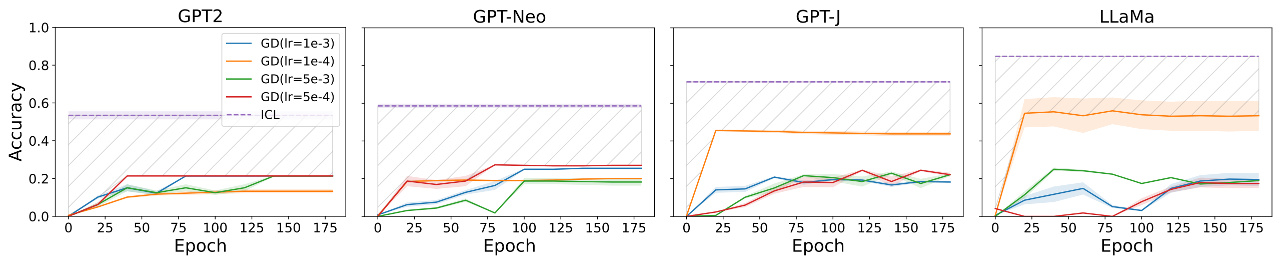


(c) *Overlap Cosine Similarity* comparison

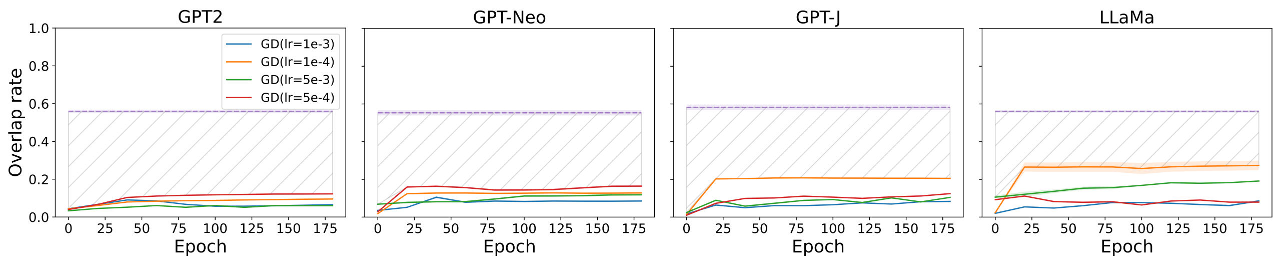
Figure 22: Comparison of ICL and $\widehat{\text{GD}}$ for the RTE dataset, with increasing number of demonstrations. $\widehat{\text{GD}}$ is simulated by optimizing on one random middle layer of LLaMa.

F. Impact of model capacity on the ICL vs GD.

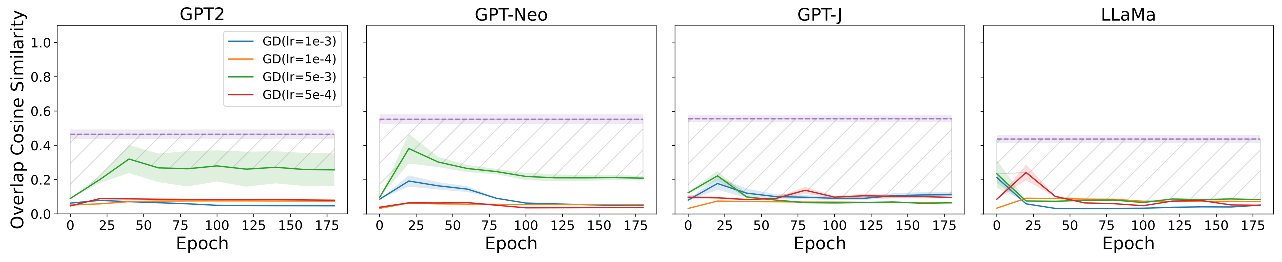
We also investigated the influence of model size on the gap between ICL and GD. Specifically, we fix the dataset to AGNews, $N = 8$, and select GPT2-XL (Radford et al., 2019), GPT-NEO (Black et al., 2021), GPT-J (Wang & Komatsuzaki, 2021) as models of choice to conduct ICL vs GD experiments. Note that the model capacity is ranked as follows: LLAMA (7B) >GPT-J (6B)>GPT-NEO (2.7B)>GPT2-XL (1.5B). The results are shown in Figure 23, from where we can see that the gap does not change significantly as the model size increases from GPT2-XL to LLAMA.



(a) *Accuracy* comparison



(b) *Token overlap* comparison



(c) *Overlap Cosine Similarity* comparison

Figure 23: Comparison of ICL and GD for the AGNews dataset as model size varies.