Understanding In-Context Learning in Large Language Models as Gradient Descent Revisted

Tomer Bar Natan, Gilad Deutch, Nadav Magar
Supervised by Guy Dar
Tel-Aviv University

Abstract

In-context learning (ICL) has shown impressive results in few-shot learning tasks, yet the underlying mechanism of it is still not fully understood. Recent research suggests that ICL is similar to gradient descent (GD)-based fine-tuning. Mathematical analysis shows an equivalence between ICL to gradient descent on a transformer with linear attention and one layer, yet empirical research has shown only limited success in showing equivalence on non-linear, multi-layered models. In this work, we experiment with a new training process that we call per-layer training, and provide some empricial evidence to suggest that this process may explain the underlying mechanism of ICL better than standard fine-tuning. Per-layer training, we beileve, aligns better with the mathematical analysis of the ICL and GD eqvuivalence. We also experiment with fine-tuning the model with linear approximation of it. Code implementation for all our experiments can be found here: https://github.com/GiilDe/ft-vs-icl

1 Introduction

In recent studies (Von Oswald et al., 2023; Dai et al., 2023), attempts have been made to establish a link between in-context learning (ICL) and the fine-tuning process using gradient descent (GD) in transformer models. One common conclusion that researchers frequently derive from these studies is that "ICL essentially enacts gradient descent." Nevertheless, it's worth noting that much of the analysis in these investigations focused on models with linear attention, which can be considered as a simplified setting.

In their study, (Dai et al., 2023) expand their findings from linear attention to conventional attention mechanisms, relying on empirical evidence to support their assertions. Their experiments convincingly demonstrate that a model fine-tuned through gradient descent steps and a model prompted with in-context examples appear to execute similar functions. In simpler terms, they exhibit analogous behaviors when processing inputs. Furthermore, they observe a substantial overlap in the internal behaviors of these two models. Yet, it remains unclear how a transformer's forward pass calculates its backward pass, even if it's done in a clever way involving a smaller transformer embedded within the weights of the original transformer, as suggested by some. Even if this is theoretically achievable, it's a complex concept to put into practice. In contrast, when it comes to linear models, the gradient step has a straightforward mathematical formula, making it much simpler to handle.

In (Von Oswald et al., 2023), they put forth a less ambitious idea. They suggest that the model essentially carries out a kind of gradient descent on a simple linear model applied to the original deep representations calculated during the forward pass. Using linear models with detailed feature representations is known to be highly effective in accomplishing various tasks.

In this study, our goal is to demonstrate that this principle holds true in this context as well. This can support our argument that extensive fine-tuning is not required, and a basic linear model that uses the original hidden states as inputs suffices. To do so, we perform three experiments:

- **Linearization**: We replicated the similarities results of (Dai et al., 2023) and compared them with a simplified version the function we optimize by linearization (Ortiz-Jimenez et al., 2023).
- **Per Layer Training**: We trained the GPT model while detaching the output of each layer from the computational graph, meaning that each layer does not propagate the gradient back to previous layers.
- **Labels Switching**: To further confirm the results of (Dai et al., 2023), motivated by

textbfTODO: add relevant labels experiments papers, we evaluated the change in similarity when providing false labels both in FT and ICL.

2 Background and Preliminaries

2.1 Dual Form Between Attention and Linear Layers Optimized by Gradient Descent

The view of language models as meta-optimizers originates from the presentation of the dual and primal forms of the perceptron (Aizerman et al., 2019). This notion was later expressed in terms key/value/query-attention operation by by (Irie et al., 2022; Dai et al., 2023; Von Oswald et al., 2023) which apply it apply it in the modern context of deep neural networks. They show that linear layers optimized by gradient descent have a dual representation as linear attention.

Let $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ be the weight matrix of a linear layer initialized at W_0, and let $\mathbf{x}, \mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^{d_{\text{in}}}$ be the input and training examples representation respectively. One step of gradient descent on the loss function $\mathcal{L}$ with learning rate η yields the weight update ΔW. This update can be written as the outer products of the training examples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and the gradients of their corresponding outputs $\mathbf{e}_i = -\eta \nabla_{W_0 x_i} \mathcal{L}$

$$\Delta W = \sum_i \mathbf{e}_i \otimes \mathbf{x}_i'^T \tag{1}$$

Thus the computation of the optimized linear layer can be formulated as

$$\begin{aligned} \mathcal{F}(\mathbf{x}) &= (W_0 + \Delta W)\,\mathbf{x} \\ &= W_0\mathbf{x} + \Delta W \mathbf{x} \\ &= W_0\mathbf{x} + \sum_i (\mathbf{e}_i \otimes \mathbf{x}_i)\,\mathbf{x} \\ &= W_0\mathbf{x} + \sum_i \mathbf{e}_i \left(\mathbf{x}_i^T \mathbf{x}\right) \\ &= W_0\mathbf{x} + \text{LinearAttn}\left(E, X, \mathbf{x}\right), \end{aligned} \tag{2}$$

where $\text{LinearAttn}(V, K, \mathbf{q})$ denotes the linear attention operation. From the perspective of attention we regard training examples X as keys, their corresponding gradients as values, and the current input $\mathbf{x}$ as the query.

2.2 Understanding Transformer Attention as Meta-Optimization

In this section we explain the simplified mathematical view of in-context learning as a process of meta-optimization presented in (Dai et al., 2023). For the purpose of analysis, it is useful to view the change to the output induced by attention to the demonstration tokens as equivalent parameter update ΔW_{ICL} that take effect on the original attention parameters.

Let $\mathbf{x} \in \mathbb{R}^d$ be the input representation of a query token t, and $\mathbf{q} = W_Q \mathbf{x} \in \mathbb{R}^{d'}$ be the attention query vector. We use the relaxed linear attention model, whereby the softmax operation and the scaling factor are omitted:

$$\begin{aligned} \mathcal{F}_{\text{ICL}}(\mathbf{q}) &= \text{LinearAttn}(V, K, \mathbf{q}) \\ &= W_V[X'; X] \left(W_K[X'; X]\right)^T \mathbf{q} \end{aligned} \tag{3}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d' \times d}$ are the projection matrices for computing the attention queries, keys, and values, respectively; X denotes the input representations of query tokens before t; X' denotes the input representations of the demonstration tokens; and $[X'; X]$ denotes the matrix concatenation.

They define $W_{\text{ZSL}} = W_V X \left(W_K X\right)^T$ as the initial parameters of a linear layer that is updated by attention to in-context demonstrations. To see this, note that W_{ZSL} is the attention result in the zero-shot learning setting where no demonstrations are given (Equation 3). Following the reverse direction of Equation (2), you arrive at the dual form of the Transformer attention:

$$\begin{aligned} \mathcal{F}_{\text{ICL}}(\mathbf{q}) =& W_{\text{ZSL}}\mathbf{q} + \text{LinearAttn}\left(W_V X', W_K X', \mathbf{q}\right) \\ =& W_{\text{ZSL}}\mathbf{q} + \sum_i W_V \mathbf{x}_i' \left(\left(W_K \mathbf{x}_i'\right)^T \mathbf{q}\right) \\ =& W_{\text{ZSL}}\mathbf{q} + \sum_i \left(W_V \mathbf{x}_i' \otimes \left(W_K \mathbf{x}_i'\right)\right) \mathbf{q} \\ =& W_{\text{ZSL}}\mathbf{q} + \Delta W_{\text{ICL}}\mathbf{q} \\ =& \left(W_{\text{ZSL}} + \Delta W_{\text{ICL}}\right) \mathbf{q}. \end{aligned} \tag{4}$$

By analogy with Equation(2), we can regard $W_K \mathbf{x}_i'$ as the training examples and $W_V X'$ as their corresponding meta-gradients.

2.3 Linearization of a Model

Resent works suggest a function we optimize can significantly simplify by a method called linearization (Lee et al., 2019; Ortiz-Jimenez et al., 2023). Specifically, these works suggest that it is possible to approximate a pre-trained model in the vicinity of its initial parameters:

$$f_{\theta_0 + \delta\theta}(x) \approx f_{\theta_0}(x) + \delta\theta^{\text{T}} \nabla_{\theta_0} f(x) := f_{\delta\theta}^{\text{lin}}(x; \theta_0) \tag{5}$$

where θ_0 represents the pre-trained model's parameters, $\delta\theta$ indicates the change in parameters during fine-tuning, and x denotes an input sequence (fixed with respect to $\delta\theta$). This is a linear model across deep representations, denoted as $\phi_i(x) = \nabla_{\theta_0} f(x)$. We don't require taking gradients through the entire network, just the coefficients. We then perform the following gradient step:

$$\theta \leftarrow \theta - \eta \nabla \mathcal{L}(f^{\text{lin}}(x), y) \cdot \phi(x)$$

In our work, we underscore the importance of exploring alternative approaches that enable the implementation of functions with reduced complexity when investigating ICL. This emphasis arises from the limitations imposed on the complexity of functions attainable through the forward pass, which must align with the network's depth. Therefore, we have explored a linearized variant of the model to address this constraint

3 Experiments

3.1 Per-Layer Training

We fine-tune the GPT model with two changes: (1) we feed the output of each attention layer to the projection head and subsequently compute the cross entropy loss on it (2) we detach the output of each layer from the computational graph, meaning that each layer does not propagate the gradient back to previous layers. Figure 1 illustrates the process.

There are 3 main motivations for this experiment:

1. Direction of "information flow": in standard model training, due to the nature of backpropagation, the gradient flows from the output layer to the input layer. This is the opposite direction of information flow in the ICL process, where each layer is ignorant of the output of the subsequent layers. In the per-layer training, each layer's gradient does not depend on later layers.

2. The analysis of the similarity between ICL and fine-tuning, as shown in equations 4 and 2 is done per-layer and is not expanded to the entire model.

3. In (Dai et al., 2023), the authors show that the similarity between ICL and finetuning is higher for the final layers of the model, suggesting that our intuition about the direction of information flow may be correct.

3.2 Evaluation Datasets

We evaluated our experiments on six datasets. **SST2** (Socher et al., 2013) **SST5** (Socher et al., 2013), **MR** (Pang and Lee, 2005) and **Subj** (Pang and Lee, 2004) are four datasets for sentiment classification; **AGNews** (Zhang et al., 2015) is a topic classification dataset; and **CB** (de Marneffe et al., 2019) is used for natural language inference.

3.3 Evaluation Metrics

In the following sections we describe the evaluation metrics used to compare the behavior of ICL and finetuning. To ensure an optimal comparison, we have adopted the identical metrics as introduced in (Dai et al., 2023): We design three metrics to measure the similarity between ICL and finetuning at three different levels: the prediction level, the representation level, and the attention behavior level.

Prediction Recall From the perspective of model prediction, models with similar behavior should have aligned predictions. We measure the recall of correct ICL predictions to correct finetuning predictions. Given a set of test examples, we count the subsets of examples correctly predicted by each model: $C_{\text{ZSL}}, C_{\text{ICL}}, C_{\text{FT}}$. To compare the update each method induces to the model's prediction we subtract correct predictions made in the ZSL setting. Finally we compute the **Rec2FTP** score as: $\frac{|(C_{\text{ICL}} \cap C_{\text{FT}}) \setminus C_{\text{ZSL}}|}{|C_{\text{FT}} \setminus C_{\text{ZSL}}|}$. A higher Rec2FTP score suggests that ICL covers more correct behavior of finetuning from the perspective of the model prediction.

Attention Output Direction In the context of an attention layer's hidden state representation space within a model, we analyze the modifications made to the attention output representation (**SimAOU**).

For a given query example, let $h_X^{(l)}$ represent the normalized output representation of the last token at the l-th attention layer within setting X. The alterations introduced by ICL and finetuning in comparison to ZSL are denoted as $h_{ICL}^{(l)} - h_{ZSL}^{(l)}$ and $h^{(l)}FT - h_{ZSL}^{(l)}$, respectively. We calculate the cosine similarity between these two modifications to obtain SimAOU (ΔFT) at the l-th layer. A higher SimAOU (ΔFT) indicates that ICL is more inclined to adjust the attention output in the same direction as finetuning. For the sake of comparison, we also compute a baseline metric known as

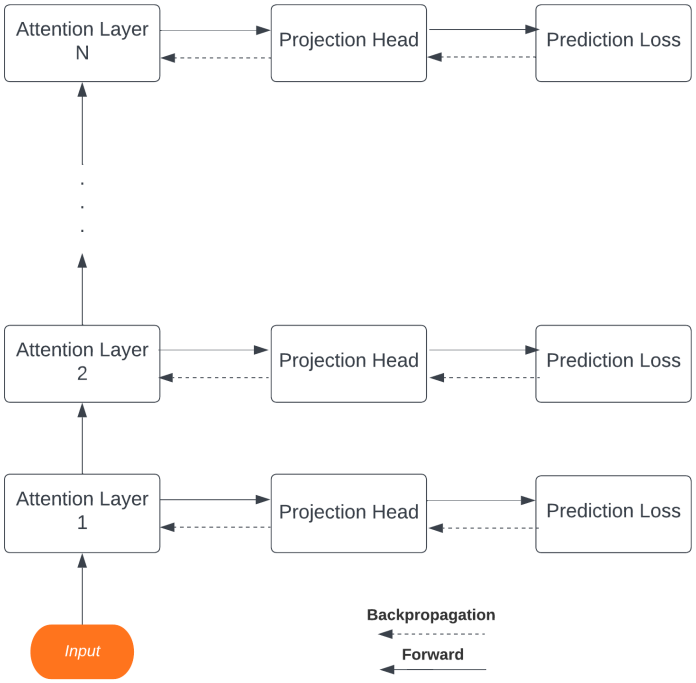

Figure 1: Per-layer training. The output of each layer is fed to the projection head and the loss is computed on it. The losses are then summed to create the loss of a single training step.

SimAOU (Random Δ), which measures the similarity between ICL updates and updates generated randomly.

Attention Map Similarity We use SimAM to measure the similarity between attention maps and query tokens for ICL and finetuning. For a query example, let $m_X^{(l,h)}$ represent the attention weights before softmax in the h-th head of the l-th layer for setting X. In ICL, we focus solely on query token attention weights, excluding demonstration tokens. Initially, before finetuning, we compute the cosine similarity between $m_{ICL}^{(l,h)}$ and $m_{ZSL}^{(l,h)}$, averaging it across attention heads to obtain SimAM (Before Finetuning) for each layer. Similarly, after finetuning, we calculate the cosine similarity between $m_{ICL}^{(l,h)}$ and $m_{FT}^{(l,h)}$ to obtain SimAM (After FT). A higher SimAM (After FT) relative to SimAM (Before FT) indicates that ICL's attention behavior aligns more with a finetuned model than a non-finetuned one.

4 Results

4.1 Linearization

We calculated SimAOU and SimAM for both original and linearized models. Results are shown in Table 1. Due to limitation in compute resources, we ran all the experiment with pretrained GPT model with 1.3 billion model parameters (GPT 1.3B). In addition we were unable evaluate the linearized model on the datasets AGNews and CB due to heavy memory consumption. It is worth mentioning that we were able to replicate the results of GPT 1.3B from (Dai et al., 2023) with a negligible difference. The results indicate that the linearized model yields lower similarity scores compared to the original model but significantly higher scores than a random vector. This suggests that our hypothesis was incorrect and other simple model versions should be thought of. More on that in the discussion section.

4.1.1 Per-Layer Training

We recreate the experiments of (Dai et al., 2023) results using the per-layer training method. The results are shown in Table 2. The SimAOU metric is higher for the per-layer training, but the SimAM metric is lower, and even lower than without finetuning for most tasks.

We hypothesize, at this point, that the reason for the lower SimAM metric is the gradient norm size. The first layers of the model are now trained on a new objective, i.e their output is projected to the vocabulary space and the cross-entropy loss

Table 1: SimAOU and SimAM on four datasets, comparing similarity between random and finetune for both original model and liniarization of the model.

Data & Metric	SST2	SST5	MR	Subj	Average
SimAOU Random	0.001	0.002	0.001	0.002	0.002
SimAOU FT	0.1091	0.113	0.219	0.193	**0.158**
SimAOU Lin Random	0.001	0.002	0.0007	0.002	0.001
SimAOU Lin FT	0.122	0.0789	0.171	0.148	0.130
SimAM Before FT	0.5547	0.3914	0.398	0.378	0.430
SimAM After FT	0.585	0.404	0.498	0.487	**0.493**
SimAM Lin Before FT	0.554	0.391	0.397	0.378	0.4305
SimAM Lin After FT	0.573	0.403	0.448	0.449	0.468

Metric \ Task	SST2	SST5	MR	Subj	AGNews	CB	Average
Sim AUO Random	0.0017	0.0029	0.001	0.0025	0.0021	0.0037	0.00231666666
Sim AUO FT	0.1091	0.113	0.219	0.193	0.3053	0.2013	0.1901166667
SimAM Before FT	0.5547	0.3914	0.398	0.378	0.1516	0.1524	0.3376833333
SimAM After FT	0.585	0.4047	0.498	0.487	0.4944	0.1875	0.4427666667

(a) original

Metric \ Task	SST2	SST5	MR	Subj	AGNews	CB	Average
Sim AUO Random	0.0016	0.0025	0.0008	0.0022	0.0021	0.0037	0.00215
Sim AUO FT	0.2297	0.1065	0.3299	0.3439	0.3213	0.3435	0.2791333333
SimAM Before FT	0.5546	0.3913	0.3979	0.3786	0.1518	0.1524	0.3377666667
SimAM After FT	0.5774	0.4039	0.2919	0.2844	0.1201	0.0293	0.2845

(b) per-layer

Table 2: Top shows the original metrics. Bottom shows the metrics for the per-layer training process. The Sim AUO metric is higher for the per-layer training.

is computed on it instead of being passed for the next numerous layers. Because of that, the gradient norm may be larger than the gradient norm of the standard training process. We verify this hypothesis by measuring the norm of the gradients of each attention layer, during the standard training process and the per-layer training process. The results are shown in Figure 2.

Following this finding, we attempt to apply gradient clipping to the per-layer training process with limited success. We experiment with different clipping values for one task (Subj) and are able to improve the metrics for this task, see Table 3. Yet, we find that each task requires a different clipping value, and aim to find a more principled approach. We attempt to clip the gradients by normalizing them to allow a maximum norm that is the norm measured in the standard fine-tuning, but this does not improve the results.

Accuracy

5 Discussions

5.1 Existence of an optimization process similar to ICL

The mathematical analysis in Section 2.2 suggests a similarity between ICL and finetuning. Yet, it's important to note its limitations: (1) it is done on a linear attention layer (2) it is done on a single layer (3) It is unclear what is the loss function for the optimization process that it suggests. This means that the analysis is not sufficient to prove that non-linear, full model ICL is indeed a form of finetuning; Hence, whether such an optimization process (or any kind of process) exists is still an open question. (3) It is unclear what is the loss function for the optimization process that it suggests. This means that the analysis is not sufficient to prove that non-linear, full model ICL is indeed a form of finetuning; Hence, whether such an optimization process (or any kind of process) exists is still an open question.

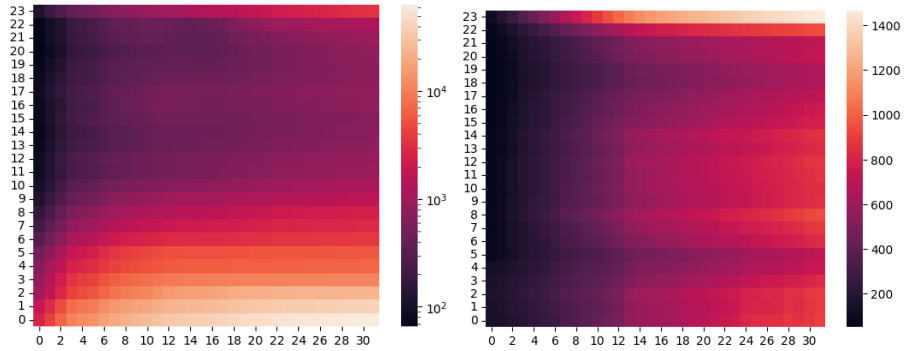

Figure 2: The gradient norm of each attention layer during the per-layer training process (left) and the standard fine-tuning process (right). The scale of the y-axis is different for each plot.

Metric \ Task	Subj
Sim AUO Random	0.0022
Sim AUO FT	0.348
SimAM Before FT	0.3786
SimAM After FT	0.4227

Table 3: Results for the per-layer training process with gradient norm clipping of max norm equal to 12.0. Compared to the results in Table 2, the SimAM metric is higher than without the clipping, but not higher the standard fine-tuning process.

6 Related Work

In-context learning (ICL) is a machine learning approach where a model fine-tunes its knowledge and adapts its behavior based on specific contextual information or examples, allowing it to perform better on tasks related to that context. It enables models to leverage domain-specific or task-specific knowledge without extensive retraining, making them more versatile and adaptable. In their work, (Brown et al., 2020) explores the remarkable ability of language models, particularly GPT-3, to learn and perform tasks with minimal examples, demonstrating their potential as versatile few-shot learners. The authors showcase the models' impressive performance across a wide range of tasks and emphasize their capacity to generalize from limited data, highlighting the transformative impact of these models on various natural language processing applications.

In recent research, there has been a growing interest in understanding the relationship between two key concepts: in-context learning (ICL) and gradient descent (GD)-based fine-tuning, particularly in the context of transformer models ((Von Oswald et al., 2023; **?**)). This research seeks to uncover how ICL, which involves adapting and learning in specific contexts, can be effectively integrated with the iterative optimization process of GD, especially when fine-tuning transformer models. However, the majority of the examination was on models that had relaxed constraints and featured linear attention mechanisms:

$$LinearAttn(K, V, q) = KV^q \tag{6}$$

The paper (Von Oswald et al., 2023), develops an explicit weight values for a linear self-attention layer, achieving an update equivalent to a single iteration of gradient descent (GD) aimed at minimizing mean squared error. Moreover, the authors demonstrate how multiple self-attention layers can progressively execute curvature adjustments, leading to enhancements over standard gradient descent. They proposed the following:

Given a 1-head linear attention layer and the tokens $e_j = (x_j, y_j)$, for $j = 1, ..., N$, one can construct key, query and value matrices W_K, W_Q, W_V as well as the projection matrix P such that a Transformer step on every token e_j is identical to the gradient-induced dynamics $e_j \to (x_j, y_j) + (0, -\delta W x_j) = (x_j, y_j) + PVK^T q_j$ such that $e_j = (x_j, y_j - \delta y_j)$. For the test data token (x_{N+1}, y_{N+1}) the dynamics are identical.

By doing so, they demonstrate the capability of linear attention to execute gradient descent on the deep representations constructed by the transformer.

Another paper ((**?**)) expand the findings from linear attention to conventional attention mechanisms, substantiating their claims with empirical data. Inspired by (Aizerman et al., 2019) and (**?**), the idea in this is paper to explain language models as meta-optimizers.

Consider W_0 and ΔW, both belonging to $\mathbb{R}^{d_{out} \times d_{in}}$, where W_0 represents the initial parameter matrix, and ΔW signifies the updating matrix. Additionally, let x be a member of $\mathbb{R}^{d_{in}}$, serving as the input representation. A linear layer, subject to optimization via gradient descent, can be articulated as follows:

$$\mathcal{F}(x) = (W_0 + \Delta W)x \tag{7}$$

In the context of the back-propagation algorithm, the determination of ΔW entails the aggregation of outer products derived from historical input representations $x'_i \in \mathbb{R}^{d_{in}}$ and their corresponding error signals $e_i \in \mathbb{R}^{d_{out}}$:

$$\Delta W = \sum_i e_i \otimes x'_i \tag{8}$$

Notably, e_i is the result of scaling historical output gradients by $-\gamma$, the negative learning rate.

By equations equation 7 and equation 8, we can derive the dual manifestation of linear layers, optimized through gradient descent, as follows:

$$\begin{aligned} \mathcal{F}(x) &= (W_0 + \Delta W)x \\ &= W_0 x + \Delta W x \\ &= W_0 x + \sum_i (e_i \otimes x'_i)x \\ &= W_0 x + \sum_i e_i (x_i'^T x) \\ &= W_0 x + \text{LinearAttn}(E, X', x) \end{aligned} \tag{9}$$

Here, E denotes historical output error signal values, X' corresponds to historical inputs employed as keys, and x serves as the current input, operating as the query.

Their experiments convincingly reveal that a model fine-tuned through gradient steps and a model prompted with in-context examples appear to perform analogous functions, exhibiting similar behaviors on inputs. Additionally, they observe significant similarities in the internal behaviors of these two models.

7 Conclusion

Following recent research that suggested a theoretical equivalency between in-context learning (ICL) and gradient descent (GD)-based fine-tuning, we've attempted to further explore this relationship in practical settings. We experimented with a new training process that we call per-layer training, and provided some empricial evidence to suggest that this process may explain the underlying mechanism of ICL better than standard fine-tuning. Another experiment that we try is to fine-tune the model with linear approximation of it. In our discussion section we discuss the limitations of our experiments, we also suggest that whether an optimization process similar to ICL exists is still an open question.

8 Acknowledge

We would like to express appreciation to Guy Dar who supervised this project, for his help with formulating the research question and methodology, give insights and continuous feedback.

References

Mark A. Aizerman, E. M. Braverman, and Lev I. Rozonoer. 2019. Theoretical foundation of potential functions method in pattern recognition.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can gpt learn in-context? language models implicitly perform gradient descent as meta-optimizers.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse.

Kazuki Irie, Róbert Csordás, and Jürgen Schmidhuber. 2022. The dual form of neural networks revisited: Connecting test time predictions to training patterns via spotlights of attention. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9639–9659. PMLR.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-

Dickstein, and Jeffrey Pennington. 2019. *Wide Neural Networks of Any Depth Evolve as Linear Models under Gradient Descent*. Curran Associates Inc., Red Hook, NY, USA.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, page 271–es, USA. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, page 115–124, USA. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 35151–35174. PMLR.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.