A Meta-Learning Perspective on Transformers for Causal Language Modeling

Xinbo Wu
University of Illinois Urbana-Champaign
xinbowu2@illinois.edu

Lav R. Varshney
University of Illinois Urbana-Champaign
varshney@illinois.edu

Abstract

The Transformer architecture has become prominent in developing large causal language models. However, mechanisms to explain its capabilities are not well understood. Focused on the training process, here we establish a meta-learning view of the Transformer architecture when trained for the causal language modeling task, by explicating an inner optimization process that may happen within the Transformer. Further, from within the inner optimization, we discover and theoretically analyze a special characteristic of the norms of learned token representations within Transformer-based causal language models. Our analysis is supported by experiments conducted on pre-trained large language models and real-world data.

1 Introduction

The Transformer architecture for neural networks is based on a self-attention mechanism (Vaswani et al., 2017) and has been widely used in natural language processing, computer vision, and scientific discovery (Devlin et al., 2019; Radford et al., 2019; Dosovitskiy et al., 2021; Vig et al., 2021), among other topical areas. Causal language modeling (CLM) aims to predict the next element in a sequence in an autoregressive manner and is one of the most important applications of the Transformer model. Large language models (LLMs) based on the Transformer architecture and CLM have shown impressive capabilities in many fields that have been considered difficult challenges in artificial intelligence (AI) (Liu et al., 2023; OpenAI, 2023). However, the underlying mechanisms of Transformer-based causal language models are still not well understood.

Meta-learning seeks to learn how to learn by leveraging common knowledge across various tasks, often through a bi-level optimization (Ravi and Larochelle, 2017; Finn et al., 2017; Franceschi et al., 2018). Within this framework, an inner process mirrors a typical model optimization process such as stochastic gradient descent (SGD) (Bottou, 2010), whereas an outer process is employed to learn hyperparameters for the inner process.

Studying Transformer models from a meta-learning perspective is an emerging research direction. Chen and Wang (2022) treat the Transformer model as a meta-learner to learn parameters for another model but did not identify a possible gradient-based inner optimization process undergone by the forward pass of the Transformer model, as we will show in the sequel. Some recent works of von Oswald et al. (2023a); Ahn et al. (2023); von Oswald et al. (2023b); Dai et al. (2023) have attempted to interpret in-context learning in Transformer-based language models from a meta-learning perspective by identifying a bi-level optimization process within in-context learning. However, they rely either on simplifications of the original Transformer model such as linear attention, or on special constructions of parameters. We make only limited assumptions to analyze a model that is as close to the Transformer used in practice as possible. However, we still do make some assumptions including zero initialization and omission of the impacts of normalization components, which makes the subject of our analysis different from Transformer models used in practice.

More importantly, rather than focusing on in-context learning that is regarded as an emergent capability of Transformer-based CLM models, we explore whether we can discover a meta-learning process within the *training* of Transformer-based CLM models. We attempt to give a plausible bi-level optimization process for a Transformer model trained for the CLM task. Additionally, to investigate the inner optimization process from the meta-learning perspective, we visualize the evolution of the token representations within Transformer-based CLM models. We uncover a characteristic of

Findings of the Association for Computational Linguistics ACL 2024, pages 15612–15622
August 11-16, 2024 ©2024 Association for Computational Linguistics

the evolution of the norms of token representations learned in Transformer-based CLMs, which may indicate a special optimization trajectory. We further theoretically analyze this characteristic using a simplified linear model. Various linear simplifications have been used in theoretical analyses of Transformer models (von Oswald et al., 2023a; Ahn et al., 2023). Although the linear model is limited in being different from practical settings, it helps us initiate the analysis of this characteristic in an otherwise complicated Transformer model.

We summarize our contributions as follows:

- We derive a plausible inner optimization process for a Transformer model trained for the CLM task and specify its objective, which is also supported by evidence from our experiments on real-world LLMs, namely GPT-2, LLaMa-7B, and LLaMa-13B.
- Building on the inner optimization process, we present a meta-learning view of a Transformer-based CLM model.
- Through experimental investigation of the inner optimization process, we discover a special characteristic of norms of the token representations learned by the Transformer-based CLM model, which may indicate a special optimization trajectory. We analyze this characteristic and further investigate it through experiments on real-world LLMs and datasets.

2 Transformer Layer may Approximate Optimization Steps on CLM

Inspired by a bi-level optimization process discovered for in-context learning, we ask whether a bi-level optimization process also exists in training a Transformer for the CLM task. Let us first formulate a linear projection weight matrix in a Transformer model by expanding it via its gradient descent learning dynamics. A weight matrix $W \in \mathbb{R}^{d_{out} \times d_{in}}$ in a Transformer model trained by standard stochastic gradient descent using T training inputs $x_1, \ldots, x_T$ to optimize a loss function $\mathcal{L}$ via backpropagation can be represented as in Irie et al. (2023):

$$W = W_0 - \sum_{t=1}^{T} \eta_t \nabla_{y_t} \mathcal{L} \otimes x_t \qquad (1)$$

where $W_0 \in \mathbb{R}^{d_{out} \times d_{in}}$ is the weight initialization, $\eta_t \in \mathbb{R}$ is the learning rate, $y_t = W_t x_t$ is the output of the linear transformation of x_t via W_t at step t and $\nabla_{y_t} \mathcal{L} \otimes x_t$ is the outer product between $\nabla_{y_t} \mathcal{L}$ and x_t. We treat $-\eta_t \nabla_{y_t} \mathcal{L} \otimes x_t$ as an error signal. Note that with our expression, training on one single token corresponds to one training step.

For simplicity, we assume W_0 is very small such that we can use a zero matrix to approximate it. Then, we have:

$$W \approx - \sum_{t=1}^{T} \eta_t \nabla_{y_t} \mathcal{L} \otimes x_t. \qquad (2)$$

The assumption of zero initialization may be better approximated by larger models. It is a common practice to initialize weights with smaller values for larger dimensions to control the initial variance of the output, thereby fostering a stable learning process and mitigating problems like exploding gradients. Numerous initialization methods commonly employed, such as Xavier (Glorot and Bengio, 2010) and Kaiming (He et al., 2015), incorporate a scaling factor proportional to the inverse of the number of dimensions in either or both the input and output.

A Transformer model is built from several Transformer layers. Each layer consists of a multi-head self-attention (MHSA) sublayer and a feedforward network (FFN) sublayer. We investigate them in the following sections.

2.1 Multi-head Self-attention may Approximate an Optimization Step

Next, we assume the dimension of a token representation within a Transformer model to be d_{model}. By following Vaswani et al. (2017), we write and expand the formulation of a multi-head attention module for the CLM task with $z \in \mathbb{R}^{n \times d_{model}}$ being the representations of current n sequential tokens (we omit the layer l of each component in the following formulations for simplicity):

$$\text{MHSA}(z_{1:n}) = \sum_{h=1}^{H} W_O^h \sum_{i=1}^{n} v_i^h \text{softmax}((k^h)^\top q_n^h)_i \qquad (3)$$

where MHSA is the parameterized multi-head self-attention module. Here, $W_O^h \in \mathbb{R}^{d_{model} \times d_{head}}$, W_V^h, W_K^h, and $W_Q^h \in \mathbb{R}^{d_{head} \times d_{model}}$ are output, value, key, and query projection matrices of head h respectively. We have $d_{model} = H d_{head}$ and H as the total number of heads. The ith value v_i^h, key k_i^h, and query q_i^h of head h are computed by $v_i^h = W_V^h z_i$, $k_i^h = W_K^h z_i$ and $q_i^h = W_Q^h z_i$ respectively. Further,

$k^h = \left[k_1^h, \dots, k_n^h\right]$ is the key matrix of head h. The ith element of the softmax output is denoted $\text{softmax}((k^h)^\top q_n^h)_i$. Throughout, we ignore the scalar factor within the softmax attention and any bias terms, to simplify analysis.

We expand W_O^h by (2) to get an approximation:

$$\text{MHSA}(z_{1:n}) \approx \sum_{h=1}^{H}(-\sum_{t=1}^{T}\eta_t \nabla_{\hat{y}_t^h}\mathcal{L} \otimes \hat{v}_t^h) \\ \sum_{i=1}^{n} v_i^h \text{softmax}((k^h)^\top q_n^h)_i \tag{4}$$

where $\mathcal{L}$ is assumed to be the CLM loss throughout and we use $\hat{\cdot}$ to denote something in the training history. Here, $\hat{v}_t^h$ refers to a historical value vector of head h at step t in the training history. Note that we omit the token positions for anything in the training history and t should not be confused with a token position within a sequence. As before, $\hat{y}_t^h = W_{O,t}^h \hat{v}_t^h \in \mathbb{R}^{d_{model}}$ and η_t^h is the learning rate of the corresponding weight matrix $W_{O,t}^{lh}$ of head h at step t.

Furthermore, by the chain rule, we have:

$$\nabla_{\hat{y}_t^h}\mathcal{L} = \left(\frac{\partial \hat{z}_t}{\partial \hat{y}_t^h}\right)^\top \nabla_{\hat{z}_t}\mathcal{L} \tag{5}$$

where $\hat{z}_t$ is a historical token representation at step t.

With (5), we can rewrite (4) as:

$$\begin{aligned}\text{MHSA}(z_{1:n}) &\approx \sum_{h=1}^{H}\left[-\sum_{t=1}^{T}(\frac{\partial \hat{z}_t}{\partial \hat{y}_t^h})^\top \eta_t \nabla_{\hat{z}_t}\mathcal{L} \otimes \hat{v}_t^h\right] \\ &\qquad \sum_{i=1}^{n} v_i^h \text{softmax}((k^h)^\top q_n^h)_i \\ &= -\sum_{t=1}^{T}\eta_t \sum_{h=1}^{H} w_t^h A_t^h \nabla_{\hat{z}_t}\mathcal{L}\end{aligned} \tag{6}$$

where w_t^h represents a weighting factor with $w_t^h = (\hat{v}_t^h)^\top \sum_{i=1}^n v_i^h \text{softmax}((k^h)^\top q_n^h)_i$ and $A_t^h = (\frac{\partial \hat{z}_t}{\partial \hat{y}_t^h})^\top \in \mathbb{R}^{d_{model} \times d_{model}}$ is a transformation matrix.

We can interpret the formulation in (6) as a key-query retrieval of a value via the softmax self-attention and then, using the retrieved value as a query and the historical values as keys to obtain a weighted average of transformed historical gradients with respect to the historical inputs to the module. In principle, the weighted average of transformed gradients could implement an approximation of the transformed gradient with respect to the current input z_n by its neighborhood in a soft manner with closeness measured by the weighting factor. This multi-retrieval process is conducted for different heads and the final error signal is also aggregated over different heads. Many methods treat a transformed gradient as a pre-conditioned gradient that captures curvature information (Park and Oliva, 2019; Martens and Grosse, 2015; Grosse and Martens, 2016), such as Newton's method for second-order optimization (Nocedal and Wright, 2006).

We can represent the nth token representation after the MHSA module with residual connection as follows:

$$\begin{aligned} z_n &\leftarrow z_n + \text{MHSA}(z_{1:n}) \\ &\approx z_n - \sum_{t=1}^{T}\eta_t \sum_{h=1}^{H} w_t^h A_t^h \nabla_{\hat{z}_t}\mathcal{L} \\ &= z_n - A\bar{\nabla}_{z_n}\mathcal{L}\end{aligned} \tag{7}$$

where we exclude normalization components such as LayerNorm (Ba et al., 2019) or RMSNorm (Zhang and Sennrich, 2019) to isolate the core mechanisms and $A\bar{\nabla}_{z_n}\mathcal{L}$ is an approximation of the transformed gradient with respect to the current input z_n based on our previous interpretation.

From (7), we can see an implementation of an approximation of transformed gradient updates to the current token representation for optimizing the CLM loss $\mathcal{L}$ is possible from the mathematical formulations.

2.2 Feed-forward Network may Approximate an Optimization Step

We can perform a similar analysis of the feed-forward network (FFN) layer. We define the FFN module as follows:

$$\text{FFN}(z_n) = W_2 \phi(W_1 z_n) \tag{8}$$

where $W_2 \in \mathbb{R}^{d_{model} \times d_{ff}}$ and $W_1 \in \mathbb{R}^{d_{ff} \times d_{model}}$ are weight matrices of the first and second layer in the FFN module and ϕ is a nonlinear activation function. Some popular choices of nonlinear activation functions are ReLU (Nair and Hinton, 2010), GELU (Hendrycks and Gimpel, 2023), and SwiGLU (Shazeer, 2020).

We can expand the formulation in (8) via (2) and chain rule:

$$\begin{aligned}\mathrm{FFN}(z_n) &\approx -(\sum_{t=1}^{T} \eta_t \nabla_{\hat{b}_t} \mathcal{L} \otimes \hat{a}_t) a_n \\ &= -\left[\sum_{t=1}^{T} (\frac{\partial \hat{z}_n}{\partial \hat{b}_t})^{\mathsf{T}} \eta_t \nabla_{\hat{z}_t} \mathcal{L} \otimes \hat{a}_t\right] a_n\end{aligned} \tag{9a}$$

$$\Rightarrow z_n \leftarrow z_n + \mathrm{FFN}(z_n)$$

$$\approx z_n - \left[\sum_{t=1}^{T} \eta_t (\frac{\partial \hat{z}_t}{\partial \hat{b}_t})^{\mathsf{T}} \nabla_{\hat{z}_t} \mathcal{L} \otimes \hat{a}_t\right] a_n \tag{9b}$$

$$\begin{aligned}&= z_n - \sum_{t=1}^{T} \eta_t w_t B_t \nabla_{\hat{z}_t} \mathcal{L} \\ &= z_n - B \bar{\nabla}_{z_n} \mathcal{L}\end{aligned} \tag{9c}$$

Here $a_n = \phi(W_1 z_n)$ and $\mathcal{L}$ is the CLM loss, $\hat{a}_t$ is a historical output of the first layer at step t. Further, $\hat{b}_t = W_{2,t}\hat{a}_t \in \mathbb{R}^{d_{model}}$ and η_t is the corresponding learning rate at step t, $w_t = (\hat{a}_t)^{\mathsf{T}} a_n$ is a weighting factor, $B_t = (\frac{\partial \hat{z}_t}{\partial \hat{b}_t})^{\mathsf{T}} \in \mathbb{R}^{d_{model} \times d_{model}}$ is a transformation matrix, and $B\bar{\nabla}_{z_n}\mathcal{L}$ is an approximation of the transformed gradient with respect to the current input z_n.

Similar to the previous analysis, we can interpret the FFN module as a query-key retrieval of a weighted average of transformed historical gradients with respect to the historical module input. In principle, the FFN module with the residual connection could approximate a transformed gradient descent update to the current token representation for optimizing the CLM loss $\mathcal{L}$.

Overall from its mathematical formulations, a Transformer layer can be understood as approximately performing two types of transformed gradient descent updates via the MHSA and the FFN modules, respectively. Particularly, the token representations are normalized after each type of transformed gradient descent update. Thus, based on this understanding, the forward pass of a series of Transformer layers trained for the CLM task minimizes an approximate CLM loss $\bar{\mathcal{L}}$ via an inner optimization process. So far, we have shown the plausibility of an inner optimization process within a Transformer trained for the CLM task from its mathematical formulations. This can serve as a first step toward our meta-learning perspective, which will be discussed in the following section. In Section 5.2, we will present empirical evidence based on real-world LLMs and data that supports this understanding.

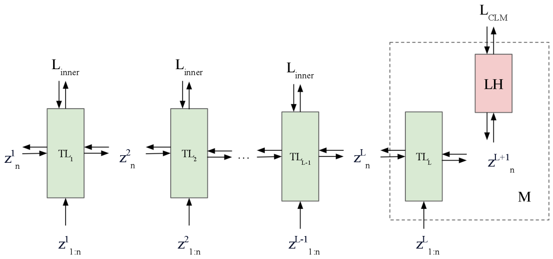


Figure 1: A meta-learning view of a Transformer model trained for CLM task. z_i^l is the ith token representation of the lth layer. TL means a Transformer layer, LH denotes a language head and M represents a predictor model. L_{inner} and L_{CLM} are an inner optimization loss and a CLM loss respectively.

3 A Meta-Learning View of Transformer

We generally view the meta-learning process as a "learning to learn" process based on a variety of tasks. The "learning to learn" process is usually formulated as a bi-level optimization process. We have discovered an optimization process inner to the Transformer's training process within its forward process, which forms a bi-level optimization process. The language data utilized for the CLM task typically spans a broad range of topics, inherently encompassing various tasks. It has long been recognized that a language model trained on such data can proficiently undertake multiple tasks. (Radford et al., 2019).

From our perspective illustrated in Figure 1, a Transformer layer takes the context tokens as the input data and uses them to form the weightings to approximate the gradients as discussed in Section 2. This can be considered as a way of learning by approximating the learning signal, the gradients from close points, different from the classical stochastic gradient descent (SGD), where the gradients are computed from a loss. Then, the layer can work as a meta-optimizer to learn to arrange the points and compute the weightings based on current data to reflect expected closeness for efficiently solving a certain task. Particularly, these close points could potentially come from different tasks such that ongoing learning could leverage knowledge from other tasks. Then, the approximate gradients are used to update the current token representation to be a better one for minimizing the CLM loss. Particularly, a Transformer layer or a meta-optimizer is trained using the CLM loss by the backpropagation from an outer optimization process. The token representations can be viewed as model parameters from an optimization perspective. Through a series of optimization steps performed by layers, the CLM loss is expected to decrease over layers. Particularly, we view the final layer along with the linear language head as a predictor model that takes a token representation as its parameter vector.

So far, we have established a meta-learning view of the Transformer trained for the CLM task. Interestingly, unlike standard gradient descents commonly used in meta-learning, we found that in principle, the Transformer could utilize a transformed gradient descent within its inner optimization process, which could be more efficient than a standard gradient descent (Nocedal and Wright, 2006). We will leave investigations of these differences to future work.

4 Optimization Trajectory

In Section 2, we developed an understanding of the forward pass of a Transformer-based CLM model as an inner optimization process. One further question to ask is whether there are any characteristics of this optimization process. A token representation is continuously updated across layers, which has a similar role to the model parameters being updated by optimization steps. An optimization trajectory can be characterized by the evolution of a token representation. Many widely used learning algorithms consist of two parts: optimization and regularization. Investigation of a single aspect won't give us a complete picture of the learning algorithms. L2-norm regularization is commonly used with SGD as a part of an optimization process. Since the token representation is the subject of optimization, this inspires us to study the property of the norm of the token representations like that of model parameters to give a more comprehensive view of the inner optimization process.

From the visualizations and quantitative analysis of the intermediate token representations in Section 5.3, surprisingly, we find the norm of a token's vector representation approximately follows an incremental trend across layers in sequence. Notice that the loss value has a decreasing trend at the same time across layers in sequence based on Section 5.2. This is interesting because it indicates the inner optimization process may follow a specific type of optimization trajectory in general instead of an arbitrary one. In this section, we perform some analysis regarding this characteristic.

We propose the following conjecture.

Conjecture 4.1 *A Transformer model trained for the causal language modeling task learns a solution such that the norm of the current token vector representation is approximately non-decreasing with increasing layer number l in the Transformer model (except for the last layer).*

We do not consider the last layer because it is a part of the predictor model from our meta-learning view discussed in Section 3. This conjecture is supported by both qualitative and quantitative analysis performed in Section 5.3. We can also study this characteristic from a vector transformation perspective. Due to the complexity of the nonlinear transformation of a Transformer layer, we initiate studies on this characteristic by using a simplified linear model. We begin by simplifying the softmax-based attention mechanism as a linear attention mechanism as done in some prior works (von Oswald et al., 2023a; Dai et al., 2023) such that the MHSA module becomes:

$$\begin{aligned}&\mathrm{MHSA}_{linear}(z_{1:n}) = \\ &\sum_{h=1}^{H} W_O^h \sum_{i=1}^{n} W_V z_i (W_K^h z_i)^\intercal W_Q^h z_n \\ &= \sum_{h=1}^{H}\sum_{i=1}^{n} W_O^h W_V z_i (z_i)^\intercal (W_K^h)^\intercal W_Q^h z_n \\ &= W_{\mathrm{MHSA}} z_n\end{aligned} \tag{10}$$

by following conventions from (3). We can now treat the MHSA module as a linear transformation using matrix W_{MHSA}.

Similarly, we obtain a linear version of the FFN module by removing its nonlinear activation function:

$$\mathrm{FFN}_{linear}(z_n) = W_2 W_1 z_n = W_{\mathrm{FFN}} z_n. \tag{11}$$

We develop a linear model corresponding to a Transformer layer by omitting the layer normalization and combining everything and residual connections:

$$z_n^{l+1} = (I + W_{\mathrm{FFN}})(I + W_{\mathrm{MHSA}}) z_n = W_{linear} z_n \tag{12}$$

where $I \in \mathbb{R}^{d_{model} \times d_{model}}$ is an identity matrix.

We view each Transformer layer as a meta-optimizer, which should be responsible for this special characteristic. Therefore, we conduct studies on the weight matrices of these meta-optimizers with a focus on their eigenvalues to see if there are any possible hidden mechanisms leading to this special characteristic. Then, we present the following proposition for the norm of the current token representations across layers in the linear model.

Proposition 1 *Let $W \in \mathbb{R}^{d_{in} \times d_{out}}$ be a linear transformation matrix, and $x \in \mathbb{R}^{d_{in}}$ and $y \in \mathbb{R}^{d_{out}}$ be the input and output of the linear transformation. We can have the Gram matrix of W decomposed by eigendecomposition as $W^{\mathsf{T}}W = U\Lambda U^{-1}$. Let $a = U^{\mathsf{T}}x$ and $b = U^{-1}x$. Given $\sum_{i=1}^{d_{out}} \lambda_i a_i b_i \geq \sum_{i=1}^{d_{out}} a_i b_i$, we have $\|y\| \geq \|x\|$, where λ_i is the ith eigenvalue of the Gram matrix.*

Proof 1 *Since $W^{\mathsf{T}}W$ is a symmetric matrix and has non-negative real eigenvalues, we can decompose $W^{\mathsf{T}}W$ by eigendecomposition: $W^{\mathsf{T}}W = U\Lambda U^{-1}$.*

$$\begin{aligned}
\|y\| &= \sqrt{y^{\mathsf{T}}y} = \sqrt{x^{\mathsf{T}}W^{\mathsf{T}}Wx} \\
&= \sqrt{x^{\mathsf{T}}U\Lambda U^{-1}x} = \sqrt{a^{\mathsf{T}}\Lambda b} \\
&= \sqrt{\sum_{i=1}^{d_{out}} \lambda_i a_i b_i} \geq \sqrt{\sum_{i=1}^{d_{out}} a_i b_i} \\
&= \sqrt{x^{\mathsf{T}}UU^{-1}x} = \sqrt{x^{\mathsf{T}}Ix} \\
&= \sqrt{x^{\mathsf{T}}x} = \|x\|
\end{aligned} \tag{13}$$

With Proposition 1, we have another conjecture.

Conjecture 4.2 *One of the intrinsic characteristics of W_{linear} in the aforementioned linear model constructed from a Transformer model trained for the CLM task is to have large enough eigenvalues for its Gram matrix such that its output has norm no less than that of its input.*

We perform experiments to verify Conjecture 4.2 in Section 5.3 with results shown in Table 2. A way to theoretically guarantee the non-decreasing property of the norm is to transform the current token representation via the linear model that we constructed, so as to meet the condition in Proposition 1.

5 Experiments

In this section, we demonstrate experimental results to provide evidence for our theoretical analysis presented in Sections 2 and 4.

5.1 Experimental Setting

Our experiments are based on three popular Transformer-based causal language models: GPT-2 (Radford et al., 2019) (released under a modified MIT license), LLaMa-7B, and LLaMa-13B (Touvron et al., 2023; Geng and Liu, 2023; Computer, 2023) (both released under a Meta license). Their parameter counts are 1.5 billion (GPT-2), 7 billion (LLaMa-7B), and 13 billion (LLaMa-13B) respectively. We use test sets of Wikitext-103 (Merity et al., 2017) and 1 Billion Words (1BW) (Chelba et al., 2013), two common language benchmark datasets in our experiments. The wikitext-103 and 1BW datasets were constructed from Wikipedia articles and news crawl data respectively, so they follow different data distributions. To have enough context for the next token prediction task, we only keep data with lengths more than or equal to 10 words. Due to limitations on computational resources, we only randomly chose 5,000 samples from the test set of 1BW. All experiments are performed using an NVIDIA A100 GPU.

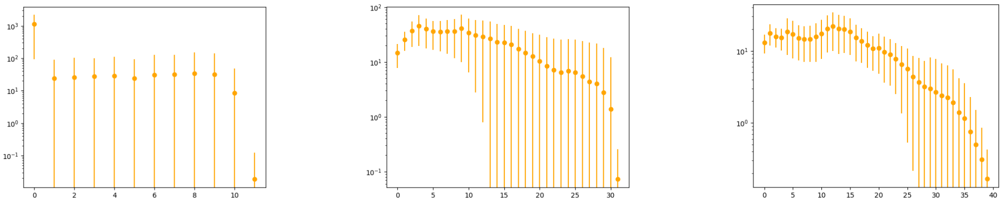
5.2 Inner Optimization Process

To investigate the possible optimization process that we discovered in our theoretical development, we perform experiments over all three models on the two datasets. We attempt to study the behavior of the inner optimization process by an approximation of the inner optimization loss with respect to the current token representation z_n^l at the lth layer as follows:

$$L_{inner,n}^{l} \approx \text{CrossEntropy}(\text{softmax}\{LH\left[TF^{L}(z_{1:n}^{l})\right]\}, y_n) \tag{14}$$

where LH is the linear head and TF^L is the last Transformer layer, together forming a predictor model, and y_n is the true label of the current token. Note that we assume TF^L only outputs the representation for the current token z_n^l. As discussed in Section 2, the inner optimization loss in our understanding is a direct approximation of the CLM loss, so using the CLM loss based on true labels would be a good approximation of the inner optimization loss of interest.

A sequence of the approximate losses for a single token is computed for a sequence of layers up to the $L-1$ layer since the Lth layer is the last layer and a part of the predictor. In this experiment, we only study the fifth or later position in a textual sequence to have enough context for the prediction task, and due to large variances, we filter out loss sequences with a final value greater than 1 to only focus on those cases in which the models are able to make a good prediction. In Figure 2, we observe



(a) GPT-2

(b) LLaMa-7B

(c) LLaMa-13B

Figure 2: Approximate Inner Optimization Loss across Layers Based on The Wikitext-103 Dataset: The x-axis is layer numbers and y-axis is the approximate loss. The mean is shown as a circle while the standard deviation is shown as a vertical bar. The vertical axis is shown in a log scale to display more subtle details.

a decreasing trend of the approximate loss on both LLaMa models across all of the datasets, mirroring an optimization process. We hypothesize the reason why GPT-2 doesn't have a obvious decreasing trend is that the zero initialization assumption isn't approximated well by a small model. We indeed observe that as the zero initialization assumptions better met by larger models, the larger models show a clearer decreasing trend. This indicates the Transformer model indeed conducts an inner optimization process for an objective similar to the CLM objective layer-by-layer and gives evidence that supports our understanding of Transformer-based CLM models proposed in Section 2. See more experimental results on the 1BW dataset in the appendix.

5.3 Studies of Token-level Representation

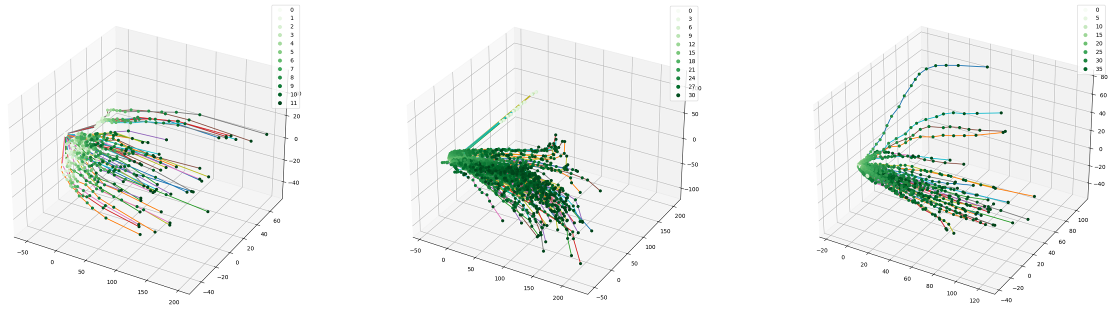
We investigate the inner optimization trajectory by studying how the current token vector representation evolves during the inner optimization process. We visualize the current token vector representations across layers in sequence for different samples in a 3D space by using principal component analysis (PCA) for dimension reduction. Note that we do not consider the last layer here because we treat it as a part of the predictor model as mentioned in Section 3. We display the optimization paths of different samples using different colors by connecting the token representations from neighboring layers. We randomly selected 50 samples for GPT-2 and 150 samples for LLaMa-7B and LLaMa-13B to make the visualizations.

Surprisingly, we observe that almost all of the samples follow a specific type of optimization trajectory instead of arbitrary ones. From Table 1, we find that for the majority of samples, the norms of their current token representations are non-decreasing along the optimization trajectory and this characteristic is also not hard to see in the visualizations in Figure 3. To study this characteristic, we measure the percentage of the optimization trajectories with non-decreasing norms (sequence-level) and the percentage of pairs of token representations at neighboring layers in a sequence whose norms are non-decreasing (pair-level). Let us assume the decreasing and non-decreasing happen uniformly at random: then the probability for a sequence of n layers to have such non-decreasing property is 2^{-n}. Considering $n = 11$ for GPT-2, $n = 31$ for LLaMa-7B, and $n = 39$ for LLaMa-13B, the chance is exceedingly small. Across different models and datasets, we consistently observe a high percentage at the sequence level compared to the null random ensemble. Moreover, the pair-level percentage is almost perfect. This gives strong evidence for our Conjecture 4.1. We present additional experimental results on the 1BW dataset in the appendix. Based on the theoretical analysis in Section 4, we construct the linear transformation matrices for GPT-2 for different layers and samples. From Table 2, the eigenvalues of the constructed matrices perfectly meet the condition proposed in Proposition 1, which means the matrix can effectively stretch the current vector representation resulting in an incremental norm. Note that we only conduct this experiment on GPT-2, because LLaMa uses a special FFN module with a gating mechanism instead of a standard one.

6 Related Work

6.1 Meta-Learning

Meta-learning aims to acquire the skill of learning itself by tapping into shared knowledge among different tasks, typically through a process involving bi-level optimization. Many meta-learning methods have shown their great efficiency in learning from few-shot examples (Ravi and Larochelle, 2017; Finn et al., 2017; Franceschi et al., 2018).



(a) GPT-2

(b) LLaMa-7B

(c) LLaMa-13B

Figure 3: Visuzalizations of Current Token Vector Representations for Different Models on The Wikitext-103 Dataset.

Model	**Wikitext-103**		**1BW**	
	Sequence (%)	Pair (%)	Sequence (%)	Pair (%)
GPT-2	92.4	99.3	83.2	98.4
LLaMa-7B	86.0	98.2	81.0	98.7
LLaMa-13B	79.8	99.0	73.4	99.0

Table 1: Sequence-level and Pair-level Measurements of The Non-decreasing Norm Characteristic: Sequence means sequence-level measurement and pair refers to pair-level measurement.

Model	**Wikitext-103**	**1BW**
GPT-2	100.0 %	100.0 %

Table 2: Percentages of Eigenvalues of Constructed Matrices Meeting The Condition in Proposition 1.

Our work focuses on studying Transformer-based causal language models from a meta-learning perspective instead of developing a new meta-learning method.

6.2 Mechanistic Interpretability

Mechanistic interpretability involves analyzing the weights of a trained neural network to reverse engineer the algorithms learned by the model (Meng et al., 2022; Elhage et al., 2021; Nanda et al., 2023; Ilharco et al., 2022; Conneau et al., 2018). Our research also endeavors to explore the hidden mechanisms of Transformer models, albeit from a distinct perspective compared to existing studies in this field.

6.3 Representations of Transformer Models

Many works have attempted to study various properties of internal representations learned by Transformer models (Thompson and Mimno, 2020; Chen et al., 2021; Reif et al., 2019). Even though some works (Fayyaz et al., 2021; Tenney et al., 2019; Niu et al., 2022) have shown increasing prediction power across Transformer layers, especially following a BERT architecture (Devlin et al., 2019), our work attempts to provide reasons for this phenomenon instead of simply presenting the existence of it. Our work also studies a special characteristic about norms of the token representations learned by the Transformer-based CLM model, which has not been explored by existing works.

7 Conclusion

In this work, we show the plausibility of an inner optimization process in a Transformer model trained for the CLM task by both mathematical derivations and empirical evidence. We also provide a meta-learning view of the Transformer model trained for the CLM task, which may provide useful insights into the learning dynamics of Transformer-based causal language models. In addition, from our experiments on the inner optimization process, we discover a special characteristic about norms of the token representations learned by the Transformer-based CLM model and investigate this characteristic by theoretical analysis and experiments on real-world large language models. Our overarching objective in this study is to provide a new perspective for studying Transformer-based CLM models as well as some empirical evidence to encourage further research in this direction. The perspective may also give more insights into the design and training of more advanced Transformer models by having a clearer view of their internal mechanism.

8 Limitations

This work is limited to enhancing the understanding of the hidden mechanisms of Transformer-based CLMs. More effort can be made by inspiration from these understandings to improve existing models and propose more advanced algorithms. Our current study verifies our discovered mechanisms of Transformer-based CLMs on limited models and datasets. It is interesting to study these mechanisms in more diverse scenarios.

References

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. 2023. Transformers learn to implement preconditioned gradient descent for in-context learning. *arXiv:2306.00297*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2019. Layer normalization. *arXiv:1607.06450*.

Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT)*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv:1312.3005*.

Boli Chen, Yao Fu, Guangwei Xu, Pengjun Xie, Chuanqi Tan, Mosha Chen, and Liping Jing. 2021. Probing bert in hyperbolic spaces. *arXiv preprint arXiv:2104.03869*.

Yinbo Chen and Xiaolong Wang. 2022. Transformers as meta-learners for implicit neural representations. *European Conference on Computer Vision*.

Together Computer. 2023. Redpajama-data: An open source recipe to reproduce llama training dataset.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*.

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. *Findings of the Association for Computational Linguistics*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Xiaohua Zhai Dirk Weissenborn, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *Proceedings of the International Conference on Learning Representations*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1.

Mohsen Fayyaz, Ehsan Aghazadeh, Ali Modarressi, Hosein Mohebbi, and Mohammad Taher Pilehvar. 2021. Not all models localize linguistic knowledge in the same place: A layer-wise probing on bertoids' representations. *arXiv preprint arXiv:2109.05958*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *Proceedings of the International Conference on Machine Learning*.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. 2018. Bilevel programming for hyperparameter optimization and meta-learning. *Proceedings of the 35th International Conference on Machine Learning*, pages 1568–1577.

Xinyang Geng and Hao Liu. 2023. Openllama: An open reproduction of llama.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Roger Grosse and James Martens. 2016. A Kronecker-factored approximate Fisher matrix for convolution layers. *Proceedings of the International Conference on Machine Learning*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Dan Hendrycks and Kevin Gimpel. 2023. Gaussian error linear units (gelus). *arXiv:1606.08415*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Kazuki Irie, Róbert Csordás, and Jürgen Schmidhuber. 2023. The dual form of neural networks revisited: Connecting test time predictions to training patterns via spotlights of attention. *Proceedings of the International Conference on Machine Learning*.

Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, Zihao Wu, Lin Zhao, Dajiang Zhu, Xiang Li, Ning Qiang, Dingang Shen, Tianming Liu, and Bao Ge. 2023. Summary of ChatGPT-related research and perspective towards the future of large language models. *arXiv:2304.01852*.

James Martens and Roger Grosse. 2015. Optimizing neural networks with Kronecker-factored approximate curvature. *Proceedings of the International Conference on Machine Learning*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. *Proceedings of the International Conference on Learning Representations*.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. *Proceedings of the International Conference on Machine Learning*.

Neel Nanda, Andrew Lee, and Martin Wattenberg. 2023. Emergent linear representations in world models of self-supervised sequence models. *arXiv preprint arXiv:2309.00941*.

Jingcheng Niu, Wenjie Lu, and Gerald Penn. 2022. Does bert rediscover a classical nlp pipeline? In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3143–3153.

Jorge Nocedal and Stephen Wright. 2006. *Numerical Optimization*. Springer Science & Business, Media.

OpenAI. 2023. GPT-4 technical report. *arXiv:2304.01852*.

Eunbyung Park and Junier B. Oliva. 2019. Metacurvature. *Advances in Neural Information Processing Systems*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. *Proceedings of the International Conference on Learning Representations*.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. *Advances in Neural Information Processing Systems*, 32.

Noam Shazeer. 2020. GLU variants improve transformer. *arXiv:2002.05202*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Laure Thompson and David Mimno. 2020. Topic modeling with contextualized word representation clusters. *arXiv preprint arXiv:2010.12626*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Jesse Vig, Ali Madani, Lav R. Varshney, Caiming Xiong, Richard Socher, and Nazneen Rajani. 2021. BERTology meets biology: Interpreting attention in protein language models. *Proceedings of the International Conference on Learning Representations*.

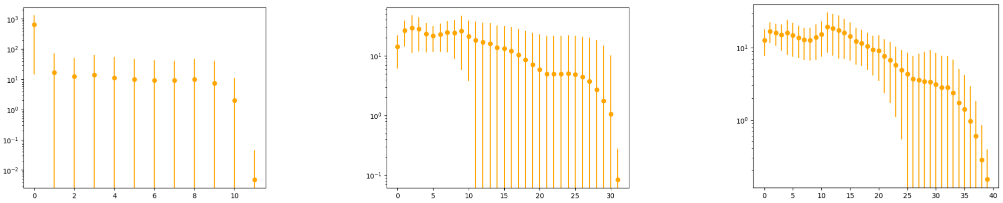
Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, Alexander Mordvintsev João Sacramento, Andrey Zhmoginov, and Max Vladymyrov. 2023a. Transformers learn in-context by gradient descent. *arXiv:2212.07677*.

Johannes von Oswald, Eyvind Niklasson, Maximilian Schlegel, Seijin Kobayashi, Nicolas Zucchet, Nino Scherrer, Nolan Miller, Mark Sandler, Blaise Agüera y Arcas, Max Vladymyrov, Razvan Pascanu, and João Sacramento. 2023b. Uncovering mesa-optimization algorithms in transformers. *arXiv:2309.05858*.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Advances in Neural Information Processing Systems*.

A Additional Experimental Results

We show additional experimental results on the inner optimization process and studies of token-level representation below.

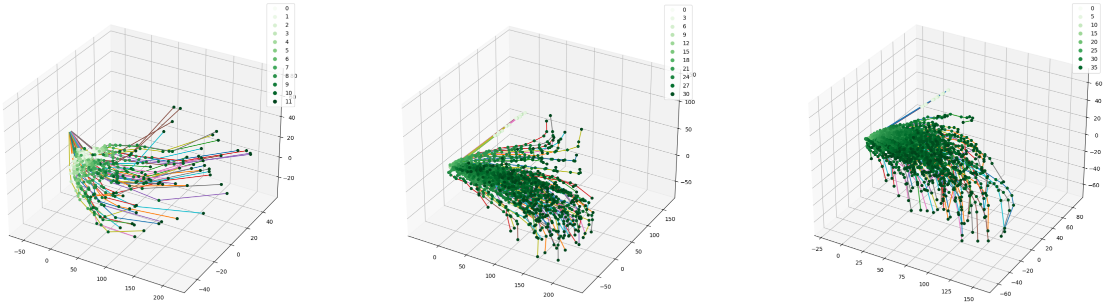


(a) GPT-2

(b) Llama-7B

(c) Llama-13B

Figure 4: Approximate Inner Optimization Loss across Layers Based on The 1BW Dataset: The x-axis is layer numbers and the y-axis is the approximate loss. The mean is shown as a circle while the standard deviation is shown as a vertical bar. The vertical axis is shown in a log scale to display more subtle details.



(a) GPT-2

(b) LLaMa-7B

(c) LLaMa-13B

Figure 5: Visuzalizations of Current Token Vector Representations for Different Models on The 1BW Dataset.