BEYOND SCALING LAWS: UNDERSTANDING TRANSFORMER PERFORMANCE WITH ASSOCIATIVE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Increasing the size of a Transformer does not always lead to enhanced performance. This phenomenon cannot be explained by the empirical scaling laws. Furthermore, improved generalization ability occurs as the model memorizes the training samples. We present a theoretical framework that sheds light on the memorization process and pre-training of transformer-based language models. We model the behavior of Transformers with associative memories using Hopfield networks, such that each transformer block effectively conducts an approximate nearest-neighbor search. Based on this, we design an energy function analogous to that in the modern continuous Hopfield network which provides an insightful explanation for the attention mechanism. Using the majorization-minimization technique, we construct a global energy function that captures the layered architecture of the Transformer. We show a dependency between the model size and the dataset for the model to attain optimal performance, and the achievable cross-entropy loss is bounded by a constant. We substantiate our theoretical findings through a series of experiments, which include tests conducted with GPT-2, vanilla Transformers, and OpenELM.

1 INTRODUCTION

Transformer-based neural networks have exhibited powerful capabilities in accomplishing a myriad of tasks such as text generation, editing, and question-answering. These models are rooted in the Transformer architecture (Vaswani et al., 2017) which employs the self-attention mechanisms to capture the context in which words appear, resulting in superior ability to handle long-range dependencies and improved training efficiency. In many cases, models with more parameters result in better performance measured by perplexity (Kaplan et al., 2020), as well as in the accuracies of end tasks (Khandelwal et al., 2019; Rae et al., 2021; Chowdhery et al., 2023). As a result, larger and larger models are being developed in the industry. Recent models (Smith et al., 2022) can reach up to 530 billion parameters, trained on hundreds of billions of tokens with more than 10K GPUs.

Nevertheless, it is not always the case that bigger models result in better performance. For example, the 2B model MiniCPM (Hu et al., 2024c) exhibits comparable capabilities to larger language models, such as Llama2-7B (Touvron et al., 2023), Mistral-7B (Jiang et al., 2023), Gemma-7B (Banks & Warkentin, 2024), and Llama-13B (Touvron et al., 2023). Moreover, as computational resources for training larger models increase, the size of available high-quality data may not keep pace. It has been documented that the generalization abilities of a range of models increase with the number of parameters and decrease when the number of training samples increases (Belkin et al., 2019; Nakkiran et al., 2021; d'Ascoli et al., 2020), indicating that generalization occurs beyond the memorization of training samples in over-parameterized neural networks (Power et al., 2022). Therefore, it is crucial to understand the convergence dynamics of training loss during memorization, both in relation to the model size and the dataset at hand. There has been an increasing interest in the empirical scaling laws under constraints on the training dataset size (Muennighoff et al., 2024). Extensive experiments have led to the conclusion of the empirical scaling laws (Kaplan et al., 2020) in terms of the model performance measured by the test cross-entropy loss. Unfortunately, this scaling law does not explain why in many cases smaller models perform better.

In this paper, we focus on the theoretical aspects of the dependencies between the achievable performance, indicated by the pre-training loss, for transformer-based models, and the model and data sizes during memorization. It has been observed that a family of large language models tends

to rely on knowledge memorized during training (Hsia et al., 2024), and the larger the models, the more they tend to encode the training data and organize the memory according to the similarity of textual context (Carlini et al., 2022; Tirumala et al., 2022). Therefore, we model the behavior of the Transformer layers with associative memory, which associates an input with a stored pattern, and inference aims to retrieve the related memories. A model for associative memory, known as the Hopfield network, was originally developed to retrieve stored binary-valued patterns based on part of the content (Amari, 1972; Hopfield, 1982). Recently, the Modern Continuous Hopfield Network (MCHN) was proposed and has been shown to exhibit equivalence to the attention mechanism (Ramsauer et al., 2020). However, the MCHN only explains an individual Transformer layer and relies heavily on regularization.

Transformer-based models consist of a stack of homogeneous layers. The attention and feed-forward layers contribute to the majority of the parameters in large models and are also the key components of the attention mechanism. Furthermore, the layered structure of the transformer networks induces a sequential optimization, reminiscent of the majorization-minimization (MM) technique (Ortega & Rheinboldt, 1970; Sun et al., 2016), which has been extensively utilized across domains such as signal processing and machine learning. Using the MM framework, we construct a global energy function tailored for the layered structure of the transformer network.

Our model provides a theoretical framework for analyzing the performance of transformer-based language models as they memorize training samples. Large language models only manifest capabilities for certain downstream tasks once the training loss reaches a specific threshold (Du et al., 2024). In practice, the training of large language models is terminated when the loss curves plateau. On the one hand, the validation loss offers valuable insights for budgetary considerations; it has been observed that even after training on up to 2T tokens, some models have yet to exhibit signs of saturation (Touvron et al., 2023). On the other hand, implementing early stopping can potentially compromise the generalization capabilities of the models (Murty et al., 2023). We conduct a series of experiments utilizing GPT-2, vanilla Transformer, and OpenELM models on various data. The experimental outcomes substantiate our theoretical results. We believe this work offers valuable theoretical perspectives on the pre-training cross-entropy loss.

Our Contribution: (1) We take a new perspective by studying Transformer behavior using associative memories with Hopfield networks. We reveal the underlying connection between the attention mechanism and nearest-neighbor search. (2) We propose an energy function for the continuous Hopfield network without the additional regularization terms. The proposed energy applies a distance metric and can be considered as a specific instance within the broader category of universal Hopfield networks. We construct a global energy function using the majorization-minimization technique, tailored for the Transformer's layered architecture. (3) Using our theoretical framework, we characterize the dependencies between pre-training loss, model size, and dataset during memorization for transformer-based language models. (4) We support our theoretical findings with empirical experiments using GPT-2, vanilla Transformers, and OpenELM models.

2 Related Work

Scaling laws. Empirical evidence suggests that the performance of models increases as both the size of the models and the volume of training data scale up (Kaplan et al., 2020; Khandelwal et al., 2019; Rae et al., 2021; Chowdhery et al., 2023). Intensive experiments on transformer-based large language models have also been conducted to explore neural scaling laws under various conditions, including constraints on computational budget (Hoffmann et al., 2022b), data (Muennighoff et al., 2024), and instances of over-training (Gadre et al., 2024). In these analyses, a decomposition of the expected risk is utilized, leading to the following fit:

$$\hat{L}(N, D) = E + \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}}, \tag{1}$$

where N and D denote the number of parameters of the model and the size of the training data respectively. For Chinchilla models, the fitted parameters are (Hoffmann et al., 2022a)

$$\alpha = 0.34, \quad \beta = 0.28, \quad E = 1.61, \quad A = 406.4, \quad B = 410.7.$$

A line of research concerns the generalization of over-parameterized neural networks (Belkin et al., 2019; Nakkiran et al., 2021; Power et al., 2022). Recent experiments show that over-trained Trans-

formers exhibit inverted U-shaped scaling behavior (Murty et al., 2023), which is not explained by the empirical scaling laws. Further discussions on the relationship between our method and the Chinchilla scaling laws are deferred to Appendix A.

Energy-based models. Energy-based models (LeCun et al., 2006), motivated by statistical physics, have become a fundamental modeling tool in various fields of machine learning over the past few decades. The central idea is to model the neural network through a parameterized probability density function $p_\theta(x)$ for $x \in \mathbb{R}^n$ and to express the distribution in terms of a learnable energy function $E_\theta(x) : \mathbb{R}^n \mapsto \mathbb{R}$ whose parameters correspond to the model's parameters as $p_\theta(x) = \frac{\exp(-E_\theta(x))}{Z_\theta}$. Here, $Z_\theta = \int \exp(-E_\theta(x))\, \mathrm{d}x$ is the normalizing constant known as the partition function.

Hopfield models. Classical Hopfield networks (Amari, 1972; Hopfield, 1982) were introduced as paradigmatic examples of associative memory. The network's update dynamics define an energy function, whose fixed points correspond to the stored memories. An important indicator is the number of patterns that the model can memorize, known as the network's storage capacity. Modifications to the energy function (Krotov & Hopfield, 2016; Demircigil et al., 2017) result in higher storage capacities (see Table 1 in Appendix B). The original model operates on binary variables, and continuous Hopfield Networks have been developed later (Hopfield, 1984). The modern continuous Hopfield network (MCHN) (Ramsauer et al., 2020) connects the continuous formulation with the attention mechanism by introducing a specific model with a softmax activation function. Given an input (e.g., a prompt), the Hopfield layer retrieves a memory by converging to a local minimum of the energy landscape, and the update rule has a nice correspondence to the query-key-value mechanism in attention. Krotov (2021) proposes a Hierarchical Associative Memory (HAM) model with a global energy function for layered networks, as opposed to energy functions for individual layers. Since the submission of our paper, we have become aware of recent findings on associative memory. The layer-wise energy function equation 4 was introduced independently by Saha et al. within the context of clustering. Millidge et al. propose a general framework for associative memories which include similarity metrics such as Euclidean distance. Therefore, the energy function here and in (Saha et al., 2023) could both be seen as a special case of the universal Hopfield networks.

3 Model

We consider tokenized training samples $\mathcal{D} = \{s^1, s^2, \ldots, s^d\}$, where each element is a sequence of tokens whose length is bounded by a number $T_{\max} \in \mathbb{N}$. Let $\widetilde{\mathcal{D}} = \{\tilde{s}^1, \tilde{s}^2, \ldots, \tilde{s}^{d'}\}$ be the set of held-out validation samples. Details on the pre-processing of the dataset is described in Appendix E. The size $D \in \mathbb{N}$ of the dataset is proportional to the number of samples $d \in \mathbb{N}$. Let $d_{\text{emb}} \in \mathbb{N}$ be the embedding dimension of the tokens, so each input sequence has $n = T_{\max} d_{\text{emb}}$ dimensions. Let $N \in \mathbb{N}$ be the number of parameters in the attention layers and the feed-forward layers, which constitute most of the parameters in the Transformer model. Suppose there are l layers, then

$$N \approx A l d_{\text{emb}}^2 = \frac{A l d_{\text{emb}}}{T_{\max}} n, \qquad D \approx T_{\max} d. \tag{2}$$

for some constant A (see Appendix E). We use a generic distance metric $d(\cdot, \cdot)$ in the Euclidean space. In practice, this metric can usually be particularized to be the Euclidean norm.

3.1 Associative memories

We consider models trained with a causal language modeling objective. Given an input sequence of tokens $s_{\leq t} = (s_1, s_2, \ldots, s_t)$, the l-layer Transformer outputs a distribution over the next token s_{t+1}. The transformer models are trained to maximize the log-probability of the correct token s_{t+1} given $s_{\leq t}$. During pre-training, input texts are segmented into sequences of length $T_{\max}$, and the model develops the ability to generate desired content, such as predicting the correct continuations. Thus, the sequences can be viewed as patterns in the setting of *associative memories*, where stored *patterns* (e.g., sequences) can be retrieved using partial contents of the patterns. Since each prediction may have access to a varying number of preceding tokens, the sequences padded to a length of $T_{\max}$ earlier may have less amount of information for the associative memory retrieval. As demonstrated in (Svete & Cotterell, 2024; Svete et al., 2024), a Transformer can be modeled as a representation-based

n-gram model with relative small n. Consequently, only a small number of padded sequences are affected by this discrepancy due to non-uniform lengths, and it does not significantly influence the collective behavior as analyzed within the framework of statistical physics.

It has been observed that the models tend to memorize patterns from the training data (Carlini et al., 2021; Biderman et al., 2024). Empirical studies on large language models have shown that the larger the models are, the more they tend to memorize training data (Carlini et al., 2022; Tirumala et al., 2022). This memorization allows the models to learn important patterns, such as world knowledge (Hsia et al., 2024), individual words (Chang & Bergen, 2022), and linguistic structure (Chang & Bergen, 2024). In light of these findings, we make the following assumption regarding memorization. As passing the text data through an embedding layer reduces their correlation, we posit that the Transformer blocks serve to store the resulting latent representations, which are extracted from the sequences once they are embedded.

Assumption 1. *During the pre-training process, the model memorizes the (latent) training samples $\mathcal{D}$ as patterns $\{\rho^1, \rho^2, \ldots, \rho^d\}$, where $\rho^i \in \mathbb{R}^n$ for $i = 1, 2, \ldots, d$.*

To be economical with notations, we use $\mathcal{D}$ to directly address the patterns $\mathcal{D} = \{\rho^1, \rho^2, \ldots, \rho^d\}$. By memorizing the samples, we mean that the patterns are stored within the model and can be retrieved when provided with an adequate prompt. Specifically, we follow the definitions in (Ramsauer et al., 2020) for pattern storage and retrieval.

Definition 1. *For every pattern ρ^i, denote by $B_i := \{x \in \mathbb{R}^n : d(x, c_i) \leq r_i\}$ an n-ball such that $\rho^i \in B_i$. The pattern ρ^i is said to be **stored** if there exists a single fixed point $\rho^{i*} \in B_i$ to which all points $x \in B_i$ converge, and $B_i \cap B_j = \emptyset$ for $i \neq j$. Such B_i is said to be associated to the pattern ρ^i, and we denote $B_i \sim \rho^i$. The pattern ρ^i is said to be **retrieved** if the converged point is ϵ-close to the fixed point ρ^{i*}.*

Remark 1. *In the work of Saha et al., a novel collective attraction mechanism is introduced to address the challenge of partial cluster assignments using associative memory. When the intersection of clusters B_i and B_j is non-empty ($B_i \cap B_j \neq \emptyset$), analogous relaxations to those presented in Equation (7) of (Saha et al., 2023) can be made.*

Without loss of generality, we assume $c_i = \rho^i$. We also assume that the small set of held-out test samples exhibits the same patterns as those in the training set. In practice, the validation samples are randomly selected from the same dataset as the training samples, preserving the distribution.

Assumption 2. *We posit that the latent representations derived from the validation set, after being processed through an embedding layer, are stored in a manner analogous to those extracted from the training set. Specifically, for every element in the set of latent patterns $\widetilde{\mathcal{D}}$ corresponding to the validation data, there exists an B_i for some $i \in [d]$. Consequently, we assume that $\widetilde{\mathcal{D}} \subset \mathcal{D}$.*

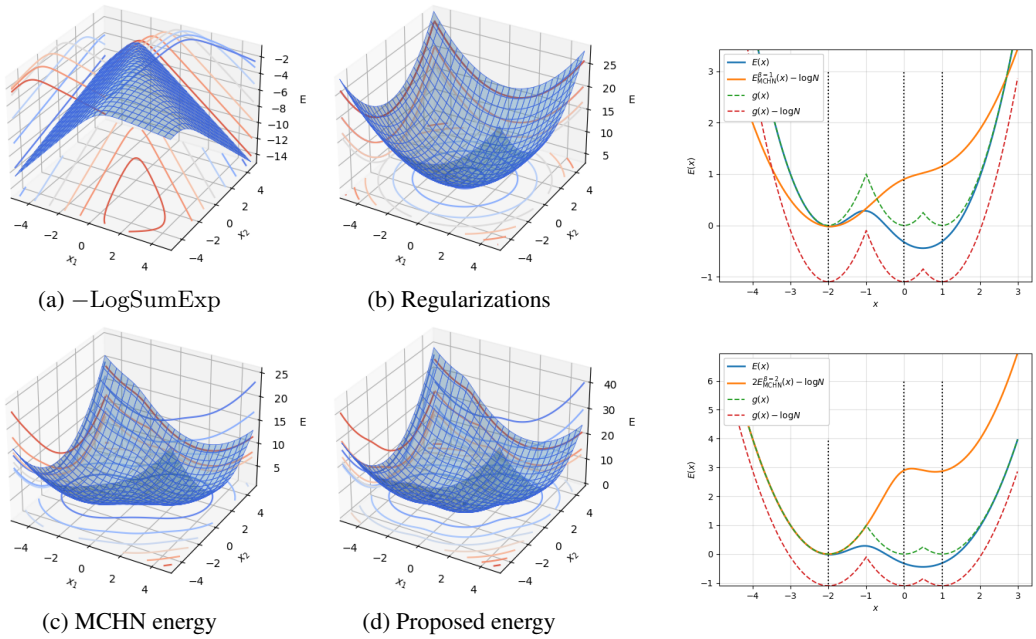
3.2 Transformer blocks

Transformer-based models, originated by Vaswani et al. (2017), are often made of a stack of homogeneous layers. The multi-head attention and feed-forward (FF) layers account for most of the parameters in the model. Appendix E provides more details using GPT-2 as an example.

Attention mechanism. The attention mechanism arguably contributes most to the overall performance of the transformer models. The attention mechanism takes three matrices $W_K \in \mathbb{R}^{d_{\text{emb}} \times d_k}$, $W_Q \in \mathbb{R}^{d_{\text{emb}} \times d_k}$, and $W_V \in \mathbb{R}^{d_{\text{emb}} \times d_v}$ as weights that can be interpreted as *keys, queries,* and *values*. Setting $d_v = d_{\text{emb}}$ facilitates the inclusion of residual connections. In a single update, an attention matrix is obtained using the update rule $\text{Attention}(Q, K, V) = V \cdot \text{softmax}(QK^\intercal/\sqrt{d_k})$.

Feed-forward layers. It has been shown that the FF layers operate essentially as key-value memories (Geva et al., 2020) such that $\text{FF}(x) = f(x \cdot K^\intercal) \cdot V$, where K, V are parameter matrices and f is a non-linear activation function such as ReLU. The FF layers can be merged into the attention without degrading the Transformer's performance (Sukhbaatar et al., 2019). Thus, the attention layer and the FF layer can be conceptually integrated into a unified transformer layer.

As the model scales, the attention and FF layers, being stacked, constitute the majority of the model's parameters. Also, since the fundamental operations of the Transformer are the attention and FF



(a) $-$LogSumExp (b) Regularizations

(c) MCHN energy (d) Proposed energy

Figure 1: **Left:** *Energy landscapes for a set of 2-dimensional patterns* $\mathcal{D} = \{(-2,-0.5),(0.2,-0.3),(1.5,1.5)\}$. (a) The negative LogSumExp function with $\beta = 1$, as an extension of (Demircigil et al., 2017). (b) The regularization terms $\frac{1}{2}x^T x + \beta^{-1}\log d + \frac{\max_i \|\rho^i\|^2}{2}$ in the MCHN energy. (c) The MCHN energy $E^1_{\mathrm{MCHN}}(x)$. (d) The proposed energy equation 4 with squared Euclidean norm. **Right:** *Energy landscapes for a set of 1-dimensional patterns* $\mathcal{D} = \{-2, 0, 1\}$. The orange curves correspond to the MCHN energy with $\beta = 1, 2$.

layers, we consider the number of parameters N in these layers, which is almost proportional to the square of the embedding dimension. The ratio depends on the number of layers and the hidden dimensions of the transformer blocks. In the current work, we do not consider other modifications such as lateral connections, skip-layer connections, mixture of experts, mixture of depths, routing, or other compressive modules such as (Xiong et al., 2023; Fei et al., 2023; Munkhdalai et al., 2024).

4 A NEW ENERGY FUNCTION

We introduce a new energy function that does not rely on additional regularization terms. We then adapt this function to the layered transformer blocks using the majorization-minimization technique. For reference, related energy functions for Hopfield networks are listed in Table 1 in Appendix B. In particular, let $M := (\rho^1, \rho^2, \ldots, \rho^d)$, the energy function for the modern continuous Hopfield network (Ramsauer et al., 2020) is

$$E^{\beta}_{\mathrm{MCHN}}(x) = -\mathrm{LogSumExp}(\beta, M^{\mathsf{T}}x) + \frac{1}{2}x^{\mathsf{T}}x + \beta^{-1}\log d + \frac{\max_i \|\rho^i\|^2}{2}, \quad \text{where}$$

$$\mathrm{LogSumExp}(\beta, y) := \beta^{-1}\log\left(\sum_{i=1}^{d}\exp(\beta y_i)\right), \quad x \in \mathbb{R}^n, \quad y = (y_1, \ldots, y_d) \in \mathbb{R}^d.$$

It can be readily observed that the negative LogSumExp function was adapted from (Demircigil et al., 2017). However, in the continuous domain, the negative LogSumExp function is not convex, making it a less suitable candidate for the energy function. The MCHN energy then adds regularization terms to create a convex energy function. These regularization terms involve both the max norm of the input and the number of patterns.

Instead of designing different regularization terms, we define a new energy function through an auxiliary function

$$g(x) := \min_{1 \le i \le d} d(x, \rho^i), \tag{3}$$

which corresponds to the *nearest neighbor search* over the the set of patterns $\mathcal{D}$. So it holds that $g(x) \geq 0$, with $g(x) = 0$ if and only if $x \in \mathcal{D} = \{\rho_1, \ldots, \rho_d\}$. According to Assumption 1, the model has memorized the patterns through pre-training; thus, the inference corresponds to a search algorithm based on some distance $d(\cdot, \cdot)$. We use the squared Euclidean 2-norm $d(x, y) = \|x - y\|^2$ in the sequel. We consider a new energy function $E(x)$ which also takes the form of LogSumExp.

$$E(x) = -\log\left(\sum_{i=1}^{d} \exp(-d(x, \rho^i))\right). \tag{4}$$

It is worth noting that the softmax function is the gradient of the LogSumExp function. So the Transformer integrates the search over layers. By summing up the negative distance between x and each stored pattern, the function assigns smaller values to points closer to the patterns. By replacing the dot product in the MCHN energy with the distance metric, $E(x)$ achieves similar goal without additional regularization. As shown in Figures 1a and 1b, as an extension of (Demircigil et al., 2017), the negative LogSumExp is not convex in the real domain, so regularization terms are applied in MCHN. Figures 1d and 1c show that the landscape of the proposed energy resembles that of the MCHN energy. In (Ramsauer et al., 2020), it is shown that E_{MCHN} induces stationary points near the stored patterns. Here, the proposed function $E(x)$ serves as a smooth surrogate of the desired function $g(x)$ in equation 3, therefore also demonstrates the retrieval ability.

Proposition 1. *Given $\mathcal{D} = \{\rho_1 \ldots, \rho_d\}$, the proposed energy $E(x)$ satisfies*

$$g(x) - \log d \leq E(x) \leq g(x).$$

The proof of Proposition 1 is due to Lemma 3 and is deferred to Appendix C. Furthermore, we show that $E(x)$ is close to the MCHN energy, as delineated below.

Proposition 2. *Let $\beta = 2$ we have*

$$|E(x) - (2E^{\beta=2}_{\text{MCHN}}(x) - \log d)| \leq \max_{1\leq i\leq d} \|\rho^i\|^2 - \min_{1\leq i\leq d} \|\rho^i\|^2.$$

The proof is given in Appendix C. Fig. 1 provides visualizations for the two propositions using low dimensional patterns. The following result follows directly from the above inequalities.

Proposition 3.

$$\min_{1\leq i\leq d} \|\rho^i\|^2 - \max_{1\leq i\leq d} \|\rho^i\|^2 \leq g(x) - 2E^{\beta=2}_{\text{MCHN}}(x) \leq \max_{1\leq i\leq d} \|\rho^i\|^2 - \min_{1\leq i\leq d} \|\rho^i\|^2 + \log d.$$

Since the proposed energy and the MCHN energy both approximate the search for the nearest pattern (desired stationary point), according to Theorem 4 in (Ramsauer et al., 2020), in each transformer layer, the probability density of the transformer layer, corresponding to the retrieval, is $p(x) = \frac{1}{Z}\exp(-E(x)|_\Omega)$, where Z is the normalizing factor, $\Omega = \bigcup_{i=1}^{d} B_i$, and B_i is as defined in Definition 1. We assume that B_i is centered at the i-th pattern. We make the following assumption on the samples. In practice, this necessitates that the training dataset exhibit a high degree of quality.

Assumption 3. *Passing the input through an embedding layer reduces the correlation between the original samples. Therefore the patterns in the latent space in $\mathcal{D}$ are well-separated, i.e., $B_i \cap B_j = \emptyset, \forall 1 \leq i < j \leq d$.*

Under Assumption 3, the energy function, confined in Ω, can be replaced by the nearest neighbor search $g(x)$. So the probability density is

$$p(x) = \frac{1}{Z}\exp(-g(x)). \tag{5}$$

4.1 The layered structure

As discussed in the related works, most Hopfield models only handle a single hidden layer, whereas SoTA transformer-based models often consist of a stack of homogeneous blocks of attention and FF layers. To model the multi-layered structure of Transformers, we employ a technique known as majorization-minimization (MM) (Ortega & Rheinboldt, 1970; Sun et al., 2016), which aims to

accelerate optimization using surrogate convex functions. We argue that the layered structure serves the same purpose when the patterns memorized by all layers encompass the set of training samples.

We divide the set of samples into $\mathcal{D} = \cup_{i=1}^{l} \mathcal{D}_i$, where $\mathcal{D}_i = \{\rho^{i_1}, \rho^{i_2}, \ldots, \rho^{i_{d_i}}\}$. Then, the energy function for each layer can be written as

$$E_t(x) = \frac{1}{Z_t} \exp(-g_t(x)), \quad \text{where} \quad g_t(x) := \min_{1 \le j \le d_t} d(x, \rho^{t_j}).$$

Denote by $x^{(0)}$ the embedding vector input into the first transformer layer and $x^{(t)} \in \mathbb{R}^n$ the output of the t-th layer for $t = 1, 2, \ldots, l$. Let $E_t(x)$ be the energy function associated with the Hopfield model of the t-th layer, then the sequential structure of the transformer network is achieved by forwarding the output $x^{(t-1)}$ to the t-th layer as input, i.e.,

$$x^{(t)} = \arg\min_{x \in \mathcal{X}_t} E_t(x), \quad \mathcal{X}_t = \{x \in \mathbb{R}^n : d(x, x^{(t-1)}) \le \delta_t\}, \qquad t = 1, \ldots, l \tag{6}$$

where the retrieved fix point attractor in the t-th layer is δ_t-close to $x^{(t-1)}$ in $d(\cdot, \cdot)$ for some $\delta_t > 0$. Such sequential optimization step is equivalent to the MM technique where every minimization step locally approximates the objective function. In particular, equation 6 corresponds to the surrogate function Eq. (3) in (Sun et al., 2016). Therefore, we define a global energy function

$$E_{\text{global}}(x) := -\text{LogSumExp}((-E_1(x), -E_2(x), \ldots, -E_l(x))). \tag{7}$$

$E_{\text{global}}(x)$ is continuous but not convex. As opposed to the HAM (Krotov, 2021), the global energy function is not a linear combination of the component energies. According to Lemma 3, we have

$$\min_{1 \le i \le l} E_i(x) - \log l \le E_{\text{global}}(x) < \min_{1 \le i \le l} E_i(x). \tag{8}$$

So $E_t(x)|_{x \in \mathcal{X}_t} \ge E_{\text{global}}(x)|_{x \in \mathcal{X}_t} + c_t$ as in Eq. (2) in (Sun et al., 2016). The probability density function corresponding to the layered transformer network can then be written as

$$p_\theta(x) = \frac{1}{Z_\theta} \exp(-E_{\text{global}}(x)), \quad x \in \Omega \tag{9}$$

where θ denotes the model's parameters and Z_θ is the normalizing constant.

5 Cross-entropy loss

We now proceed to analyze the cross-entropy loss, a metric that quantifies the divergence between predicted probabilities and actual labels, and is widely utilized for training Transformer models. The attention mechanism encompasses a softmax operation that generates a probability distribution $p \in \Delta_n$. In practice, the final softmax output is subsequently input into a task-specific layer to facilitate downstream tasks, such as predictions and classifications. The attention softmax influences the model's ability to understand and process the input data, which in turn affects the output probabilities that are used to calculate the cross-entropy loss. In essence, the attention softmax indirectly influences the cross-entropy loss by shaping the model's predictions. Consequently, we evaluate the alignment between the final softmax output of the transformer blocks and the target distribution. We demonstrate that the cross-entropy loss can be articulated through the logarithm of the partition function of the model's distribution. This formulation reveals how the allocation of attention weights is contingent upon the learned patterns, thereby establishing a relationship between the characteristics of the training data and the model size, which is pivotal for attaining optimal performance.

Let us consider the cross-entropy loss on the validation data $\widetilde{\mathcal{D}}$. Generally, the cross-entropy loss is the negative log-likelihood computed over a mini-batch. Since we are considering un-batched validation samples, the loss is normalized by the size d'. According to equation 8, there exist a layer t such that $E_{\text{global}}(x)$ is close to $E_t(x)$, i.e.,

$$E_{\text{global}}(x) = E_t(x) - \log l + c(x), \qquad c(x) \in C^\infty(\mathbb{R}^n) \tag{10}$$

such that $0 \le c(x) < \log l$. To simplify, we further assume that $c(x) = c \in [0, \log l)$ is constant. Under Assumption 1, the target distribution, which encodes all the patterns in $\mathcal{D}$, is given by

$$p_{\mathcal{D}}(x) = \sum_{i=1}^{d} p_i \delta(x - \rho^i), \quad x \in \mathbb{R}^n$$

where $\delta(\cdot)$ is the Dirac delta function such that $\delta(x) = 0, \forall x \neq 0$ and $p_i = \Pr(x = \rho^i)$ is the probability mass assigned to pattern ρ^i for $i = 1, 2, \ldots, d$. Suppose the data points are homogeneous, i.e., $p_i = \frac{1}{d}$, then $P_{\mathcal{D}}(x) = \frac{1}{d}\sum_{i=1}^{d} \delta(x - \rho^i)$, and the corresponding test samples $\widetilde{\mathcal{D}}$ induces

$$P_{\widetilde{\mathcal{D}}}(x) = \frac{1}{d'}\sum_{i=1}^{d'} \delta(x - \rho^{\sigma(i)}), \quad \sigma(\cdot) \in \mathrm{Sym}([d]). \tag{11}$$

Proposition 4. *Let L be the cross-entropy loss of the above model, then*

$$L \approx \log Z_t + \frac{1}{Z_t} \geq 1, \qquad \textit{where } \ c \in [0, \log l).$$

The proof is deferred to Appendix D.1. Note that the empirically obtained loss function equation 1 for the Chinchilla model converges to $\hat{L}(N, D) = 1.61$ as $N \to \infty$ and $D \to \infty$, which corroborates our theory that $L(N, D) \approx \log Z_t + \frac{1}{Z_t} \geq 1$, with minimum obtained when $Z_t = 1$.

Next, we explore the optimal balance between model size and data during memorization. We take a closer look at the layer partition function, which gives us

$$\begin{aligned} Z_t &= \int_{x\in\Omega} \exp(-g_t(x))\, \mathrm{d}\mu = \int_{x\in\Omega} \exp(-\min_i d(x, \rho^{t_i}))\, \mathrm{d}\mu \\ &= \sum_{i=1}^{d} \int_{x\in B_{t_i}} \exp(-\|x - \rho^{t_i}\|^2)\, \mathrm{d}x \overset{(c)}{=} 2\sum_{i=1}^{d} \pi^{\frac{n}{2}} \frac{\gamma(n, r_i)}{\Gamma(\frac{n}{2})} \end{aligned} \tag{12}$$

where (c) is due to Appendix D.2, r_i is the radius of B_i. According to Lemma 5, we have

$$e^{-r_i} V_n(r_i) \leq 2\pi^{\frac{n}{2}} \frac{\gamma(n, r_i)}{\Gamma(\frac{n}{2})} = \int_{x\in B_i} \exp(-\|x - \rho^i\|^2)\, \mathrm{d}x \leq V_n(r_i),$$

where $V_n(r) = \pi^{\frac{n}{2}} r^n / \Gamma(1 + \frac{n}{2})$ is the hyper-volume of the n-dimensional ball of radius r. Note that the volume of the unit ball $V_n(1)$ in higher dimensions decreases fast with respect to the increase in dimensionality. The stability of the probabilities is attributed to the application of normalization operators, including LayerNorm (Xiong et al., 2020) and RMSNorm (Zhang & Sennrich, 2019), which regulate the distribution of activations. The gamma function can be approximated using Stirling's approximation (Appendix D.2) for large values of its argument, which gives us

$$V_n(1) \approx \frac{\pi^{n/2}}{\sqrt{2\pi(n/2)}\left(\frac{n/2}{e}\right)^{n/2}} = \frac{1}{\sqrt{n\pi}}\left(\frac{2\pi e}{n}\right)^{\frac{n}{2}}$$

For $V_n(r)$ to have a volume of $O(1)$, the radius r must be approximately $\sqrt{n/(2\pi e)}$ asymptotically. Bringing $r = \sqrt{n/(2\pi e)}$ to equation 12, we get

$$\frac{d \cdot V_n(\sqrt{\frac{n}{2\pi e}})}{\exp(\sqrt{\frac{n}{2\pi e}})} \leq Z_t \leq d \cdot V_n(\sqrt{\frac{n}{2\pi e}}). \tag{13}$$

According to equation 2, $N \approx \frac{Ald_{\mathrm{emb}}}{T_{\max}} n$ and $D \approx T_{\max} d$. Therefore, for Z_t to reach $Z_t = 1$, we need $N = O(D^2)$ for well-separated patterns $\mathcal{D}$. The following proposition summarizes the result.

Proposition 5. *During memorization of well-separated patterns learned from the data, to minimize the cross-entropy loss, the optimal balance between model size N and data size D is $N = O(D^2)$.*

In Table 2 in Appendix B, we compare the reported cross-entropy loss of various transformer-based models in the literature. Usually, a family of models ranging in a variety of sizes is reported, and we select the largest ones. We observe that similar cross-entropy loss is achieved across a wide range of architectural shapes (including depth, width, attention heads, FF dimensions, and context lengths). Nevertheless, the pre-training cross-entropy losses all satisfy $L > 1$.

Remark 2. *We remark that some models add auxiliary regularization terms such as the z-loss (Chowdhery et al., 2023; Yang et al., 2023) during their training. In these cases, the scaling laws should take into consideration the additional terms. Also, modifications to the transformer blocks, such as additional layer normalization may contribute to the lower bound of the cross-entropy.*

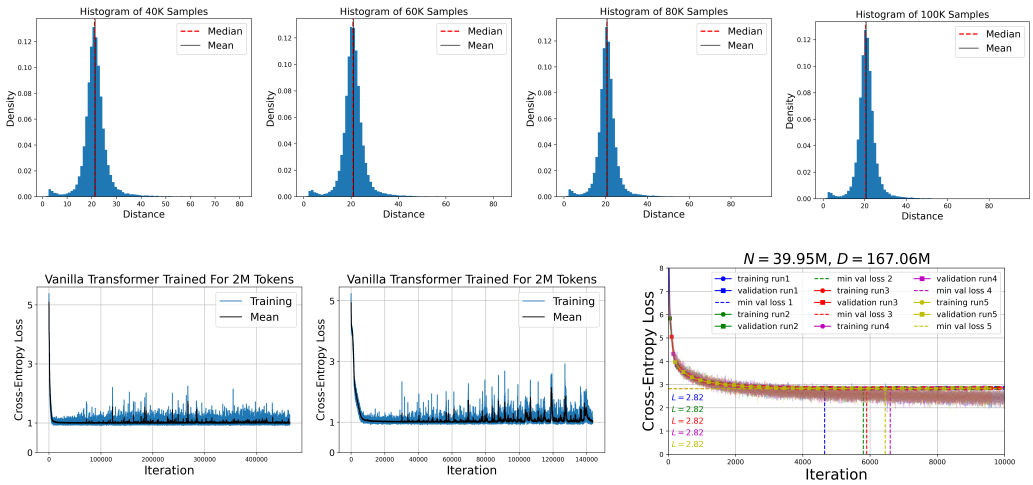


Figure 2: **Top:** Distribution of nearest neighbor distances for output activations utilizing $40\%, 60\%, 80\%$, and 100% of output data. The mean and median values of these distances consistently hover around 20, aligning closely with $2\sqrt{n/2\pi e}$ as hypothesized. **Bottom-left:** Performance of vanilla Transformers with 6 layers (*left*) and 10 layers (*middle*), each trained on the 2M Question-Formation dataset. The models were configured according to the experimental setup detailed in (Murty et al., 2023). The training losses for both models converge to a value of approximately 1, a finding that is consistent with Proposition 4. **Bottom-right:** The pre-training loss (dots) and validation loss (squares) of an OpenELM model across five training runs. The minimal validation losses are displayed in dashed lines. Each run's performance is marked by distinct colors, with the minimum validation loss value for each run indicated along the y-axis.

6 Empirical Results

We explore the hypothesis regarding the radius r in Section 5 using a pre-trained GPT-2 *medium* model. Additionally, we train vanilla Transformers and OpenELM models of different sizes to explore their cross-entropy losses.

6.1 Empirical Evaluation of the Radius

We evaluate the radius r in Z_l of a pre-trained GPT-2 *medium* model. We use the 24-layer pre-trained GPT-2 model (Radford et al., 2019)[1]. The medium size model has 355M parameters. The model is pre-trained with the next sentence prediction task on a large (40 GB) text corpus extracted from web pages. The hidden dimension is $d_{\text{emb}} = 1024$.

We test the model on the OpenWebText (Gokaslan & Cohen, 2019) dataset, a reproduction of the WebText dataset used for training the GPT-2 model. The dataset contains 9B tokens from 8,013,769 documents. We randomly sample 100K chunks of 256 tokens from the dataset. These cover approximately 1.3% of the documents and constitute approximately 0.25% of the tokens used for training. For each sample chunk, we record the activation vector of the last layer for prediction of the next token. As discussed above, each vector should be close to a stored pattern ρ^{l_i}. We calculate the distance between each activation vector and its nearest neighbor in terms of the L_2 norm and find the nearest neighbor distance for each vector, which results in 100K distance values. In Fig. 2 (top), we plot the histogram of the nearest neighbor distances for these output activations using $40\%, 60\%, 80\%$, and 100% of the output vectors. In all these cases, the mean and median equals approximately to 20, so that a typical B_i of pattern ρ^i has radius 10. This corroborates equation 13 in Section 5, according to which the radius is of order $\sqrt{1024/(2\pi e)} = 7.74$. The activation is only collected for 1.3% of the documents, so the estimated radius may be greater than the actual value.

[1]available at https://github.com/openai/gpt-2

6.2 Training vanilla Transformers

We next train vanilla Transformers using a small amount of high-quality data. The Question-Formation dataset, proposed by McCoy et al. (2020), consists of pairs of English sentences in declarative formation and their corresponding question formation. The dataset contains $D = 2$M tokens. The sentences are context-free with a vocabulary size of 68 words, and the task is to convert declarative sentences into questions.

We follow the settings in (Murty et al., 2023) to train two vanilla Transformers ($d_{\text{emb}} = 512, T_{\max} = 5000$) with $l = 6$ layers and $l = 10$ layers respectively. The training losses are shown in Fig. 2 (bottom-left), where the losses stabilize at a value of around $L = 1$ as predicted in Proposition 4.

6.3 Training models with varying widths and dimensions

Next, we train models of different sizes following the OpenELM (Mehta et al., 2024) design[2], varying the model configurations including the dimensions and the numbers of heads to achieve different model sizes while keeping the number of layers fixed. We choose different widths and dimensions such that the number of parameters of the transformer layers are about N =40M, 60M, and 80M respectively. The configurations and hyperparameters can be found in Appendix F. Specifically, the number of layers is fixed in our experiments, as empirical evidence has demonstrated that the depth is a determinant factor influencing the performance.

We utilize the Standardized Project Gutenberg Corpus (SPGC) dataset (Gerlach & Font-Clos, 2020), which contains a filtered timeseries of word-tokens without punctuation, derived from the Project Gutenberg digital library of public domain literary works. We choose this subset because it offers a collection of high-quality word sequences. We prepare the training data using different proportions of the first 180M words of the SPGC, and we use the last 5% tokens as the validation set. We pre-train the models from scratch on the tokenized training data with GPT-2 encoding, which encompasses eight distinct sizes ranging uniformly from D =167.06M to D =333.17M. Throughout the training, we employ random sampling to select chunks of the pre-determined context length. Given the typically extensive length of the e-books within this dataset, it is plausible that sequences drawn with the context length of 1024 tokens originate from the same book.

We report the number of transformer parameters N and the corresponding D^* such that training loss plateaus and the test loss is minimized, defined by $D^* = \min \{D : \text{MSE}(L_{\text{train}}^{(N,D)}, L_{\min}^{(N,D)}) < \sigma^2\}$, where $L_{\text{train}}^{(N,D)}$ is the training loss of model of size N using training data of size D, $L_{\min}^{(N,D)}$ is the minimal validation loss throughout the training steps (as is depicted in the bottom-right of Fig. 2), and MSE is the mean squared error taken over the proceeding 1000 iterations of the step where the validation loss is minimized. In Table 3, we report the MSE for each configuration, with D^* highlighted in each row when $\sigma^2 = 0.04$. Upon analyzing the optimal combinations, we have computed the ratio of model parameters to the square of the dataset size, as demonstrated in the last column. The threshold σ were empirically selected based on our preliminary experiments, as provided in Appendix F for visual inspection. Our findings indicate that the ratio N/D^{*2} consistently approaches a constant value, reinforcing our theoretical results in Section 5.

7 Conclusion

We model transformer-based networks with associative memory and study the cross-entropy loss with respect to model and data sizes. We demonstrate that the proposed energy function corresponds to a nearest neighbor search across patterns memorized during training. We then construct a global energy function for the layered structure of the transformer models using the majorization-minimization technique. We believe the current paper represents an important step towards understanding the pre-training behaviors of large transformer models. Given the significant allocation of computational resources to other scaling law investigations, we acknowledge that our numerical experiments constitute a preliminary evaluation, constrained by computational restrictions. However, it is imperative to underscore the theoretical significance of these findings.

[2]available at https://github.com/apple/corenet.

ETHICS AND REPRODUCIBILITY STATEMENT

Limitations. The theoretical results rely on the assumptions, including Assumption 1 and Assumption 3, made in the main content. We have shown that there exists a dependency between the best model size and the dataset used for training transformer-based models, both in theory and in practice. The most notable limitation is that achieving optimal performance through memorization requires high-quality data. Since most models are trained on data derived from the Internet, the resulting patterns may not be well separated. Future work will be required to identify the relationship between the dataset and the learned patterns. The experiments are conducted on GPT-2, OpenELM models, and vanilla Transformers. We expect that these results generalize to other transformer models. Also, we do not consider other modifications such as lateral connections, skip-layer connections, mixture of experts, mixture of depths, routing, or other compressive modules in the current work. These will be interesting future directions.

Broader impacts. This study elucidates the performance of transformer-based models, measured by cross-entropy loss. Consequently, the findings presented herein have the potential to influence strategic budget allocation and model lifecycle management. By offering insights into the balance between model performance and resource efficiency, it provides insights into the theoretically optimal cross-entropy loss, which can inform both budgetary planning and model termination strategies. We believe that this research delineates a constructive pathway for organizations to foster a more sustainable approach to optimize their machine learning initiatives.

Given that the patterns recognized are directly extracted from the dataset at hand, there exists a potential negative societal consequence: the propensity for models to memorize and thereby perpetuate biases against certain groups. This risk can be attenuated through the implementation of rigorous class-balancing methodologies. Nonetheless, it is imperative that comprehensive fairness assessments be conducted prior to the deployment of any model.

Reproducibility. While the present study is predominantly theoretical in its nature, we have taken multiple steps to ensure the reproducibility of our experiments. We refer the reader to Section 6 and Appendix E for a complete description of the experiments. We have also attached necessary code in the supplementary material.

REFERENCES

S-I Amari. Learning patterns and pattern sequences by self-organizing nets of threshold elements. *IEEE Transactions on Computers*, 100(11):1197–1206, 1972.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Tris Warkentin Jeanine Banks and Tris Warkentin. Gemma: Introducing new state-of-the-art open models, 2024.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Stella Biderman, Usvsn Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*, 2022.

Tyler A Chang and Benjamin K Bergen. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16, 2022.

Tyler A Chang and Benjamin K Bergen. Language model behavior: A comprehensive survey. *Computational Linguistics*, pp. 1–58, 2024.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Stéphane d'Ascoli, Levent Sagun, and Giulio Biroli. Triple descent and the two kinds of overfitting: Where & why do they appear? *Advances in Neural Information Processing Systems*, 33:3058–3069, 2020.

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168:288–299, 2017.

Zhengxiao Du, Aohan Zeng, Yuxiao Dong, and Jie Tang. Understanding emergent abilities of language models from the loss perspective. *arXiv preprint arXiv:2403.15796*, 2024.

Weizhi Fei, Xueyan Niu, Pingyi Zhou, Lu Hou, Bo Bai, Lei Deng, and Wei Han. Extending context window of large language models via semantic compression. *arXiv preprint arXiv:2312.09571*, 2023.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, et al. Language models scale reliably with over-training and on downstream tasks. *arXiv preprint arXiv:2403.08540*, 2024.

Martin Gerlach and Francesc Font-Clos. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 22(1):126, 2020.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2019.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022a.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. *Advances in Neural Information Processing Systems*, 35:30016–30030, 2022b.

Benjamin Hoover, Yuchen Liang, Bao Pham, Rameswar Panda, Hendrik Strobelt, Duen Horng Chau, Mohammed Zaki, and Dmitry Krotov. Energy transformer. *Advances in Neural Information Processing Systems*, 36, 2024.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10):3088–3092, 1984.

Jennifer Hsia, Afreen Shaikh, Zhiruo Wang, and Graham Neubig. Ragged: Towards informed design of retrieval augmented generation systems. *arXiv preprint arXiv:2403.09040*, 2024.

Jerry Yao-Chieh Hu, Pei-Hsuan Chang, Robin Luo, Hong-Yu Chen, Weijian Li, Wei-Po Wang, and Han Liu. Outlier-efficient hopfield layers for large transformer-based models. *arXiv preprint arXiv:2404.03828*, 2024a.

Jerry Yao-Chieh Hu, Donglin Yang, Dennis Wu, Chenwei Xu, Bo-Yu Chen, and Han Liu. On sparse modern hopfield model. *Advances in Neural Information Processing Systems*, 36, 2024b.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024c.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

Dmitry Krotov. Hierarchical associative memory. *arXiv preprint arXiv:2107.06446*, 2021.

Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in Neural Information Processing Systems*, 29, 2016.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and Fujie Huang. A tutorial on energy-based learning. *Predicting Structured Data*, 1(0), 2006.

Tianjin Li, Mufeng Tang, and Rafal Bogacz. Modeling recognition memory with predictive coding and hopfield networks. In *NeurIPS 2023 Workshop on Associative Memory & Hopfield Networks*.

R Thomas McCoy, Robert Frank, and Tal Linzen. Does syntax need to grow on trees? sources of hierarchical inductive bias in sequence-to-sequence networks. *Transactions of the Association for Computational Linguistics*, 8:125–140, 2020.

Sachin Mehta, Mohammad Hossein Sekhavat, Qingqing Cao, Maxwell Horton, Yanzi Jin, Chenfan Sun, Seyed Iman Mirzadeh, Mahyar Najibi, Dmitry Belenko, Peter Zatloukal, et al. OpenELM: An efficient language model family with open training and inference framework. In *Workshop on Efficient Systems for Foundation Models II@ ICML2024*, 2024.

Beren Millidge, Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Universal hopfield networks: A general framework for single-shot associative memory models. In *International Conference on Machine Learning*, pp. 15561–15583. PMLR, 2022.

Niklas Muennighoff, Alexander Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A Raffel. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*, 2024.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. Grokking of hierarchical structure in vanilla transformers. *arXiv preprint arXiv:2305.18741*, 2023.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

JM Ortega and WC Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*, volume 30. SIAM, 1970.

Bowen Peng, Jeffrey Quesnelle, Dillon Rolnick, Ari Lotter, Umer H Adil, and Esteban La Rocca. *A Preliminary report on DisTrO*, 2024. URL https://github.com/NousResearch/DisTrO/tree/main. Available at https://github.com/NousResearch/DisTrO/tree/main, date: 2024-09-10.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

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

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, et al. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2020.

Bishwajit Saha, Dmitry Krotov, Mohammed J Zaki, and Parikshit Ram. End-to-end differentiable clustering with associative memories. In *International Conference on Machine Learning*, pp. 29649–29670. PMLR, 2023.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing NLG 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*, 2019.

Ying Sun, Prabhu Babu, and Daniel P Palomar. Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Transactions on Signal Processing*, 65 (3):794–816, 2016.

Anej Svete and Ryan Cotterell. Transformers can represent n-gram language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 6841–6874, 2024.

Anej Svete, Nadav Borenstein, Mike Zhou, Isabelle Augenstein, and Ryan Cotterell. Can transformers learn n-gram language models? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9851–9867, 2024.

Mufeng Tang, Tommaso Salvatori, Beren Millidge, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Recurrent predictive coding models for associative memory employing covariance learning. *PLoS Computational Biology*, 19(4):e1010719, 2023.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290, 2022.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Dennis Wu, Jerry Yao-Chieh Hu, Teng-Yun Hsiao, and Han Liu. Uniform memory retrieval with larger capacity for modern hopfield models. *arXiv preprint arXiv:2404.03827*, 2024.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

A Discussion

Relationship to the Chinchilla Scaling Laws. Our findings are contingent upon the transformer layers' memorization of patterns, which are presumed to be well-separated points learned from the data. In our experiments, we utilize a reduced dataset to emulate the conditions of pattern separation and memorization. Moreover, we have trained the models on their respective datasets repeatedly, ensuring that the training losses have stabilized and the test losses have begun to ascend, indicative of a mild degree of over-parameterization. These conditions diverge from those of the Chinchilla experiment. In practical scenarios, commercial LLMs, akin to the Chinchilla model, are not subjected to such conditions. As we have noted in our paper, it has been observed that even after training on up to 2T tokens, some models have yet to exhibit signs of saturation. In practice, we have observed that the majority of transformer models at the commercial level tend to achieve a cross-entropy loss of approximately 2.2. The optimal balance between model and data sizes, however, is often determined by the collective expertise of practitioners. Additionally, the performance of these models can be compromised by both early and delayed stopping. Therefore, our current experimental setup represents an idealized condition that has not been encountered in commercial LLMs. Nevertheless, considering the substantial computational resources allocated to other scaling law investigations, we recognize that our numerical experiments represent a preliminary assessment that is contingent upon computational limitations, with a thorough analysis reserved for subsequent research endeavors.

Relationship to Prior Work on Hopfield Networks Recently, there has been a growing interest in physics-informed neural networks. The Energy Transformer (Hoover et al., 2024) designs a energy attention mechanism and the corresponding energy function, resulting in a unique model with a strong theoretical foundation that achieves SoTA results on graph anomaly detection and graph classification tasks. Wu et al. (2024) introduces a kernelized version of modern Hopfield networks, aiming to express energy in a feature space where patterns are well-separated, thus avoiding memory interference. Hu et al. (2024b) presents a theoretical framework for deriving and analyzing a family of modern Hopfield models, and Hu et al. (2024a) offers a compression method for Hopfield models, showing superior post-quantization performance compared to vanilla Transformers. While the existing literature, such as Hierarchical Associative Memory Krotov (2021), employs a system of differential equations to design a global energy function that can encompass feedback connections, our proposed model tackles the global energy specific to feedforward architectures. Predictive coding networks (Tang et al., 2023; Li et al.) incorporate recurrent connections; however, their dynamics are focused on minimizing the total squared prediction errors and has less connection to the attention mechanism. By conceptualizing the information retrieval properties of the Transformer as a sequence of associative memories, our model endeavors to establish relationships between model size and memorization (of training data) within the framework of statistical physics.

Summary of Notations

ρ_i	The i-th pattern
A	Constant depends on the number of layers and the hidden dimension of the network
B_i	Ball associated to the i-th pattern
d'	Number of samples in the test samples
$d(\cdot,\cdot)$	A distance metric
D	Size of the training data, $D \approx T_{\max} d$
d	Number of samples in the training samples
d_{emb}	Embedding dimension
L	Cross-entropy loss
l	Number of Transformer layers
N	Number of Transformer parameters
n	Dimension of input sequences, $n = T_{\max} d_{\text{emb}}$
R	Radius of the sphere S of patterns
$T_{\max}$	Max number of tokens in a sequence

B DEFERRED TABLES

Table 1: Table of selected related works for Hopfield network, enumerating their domain, energy function, and memory capacity. For all the works above, n represents the dimension of the input vector. W is the outer product of the patterns. M is the matrix of patterns. r is the order of polynomial $F(\cdot)$, d is the number of patterns, and c is a positive constant.

Reference	Domain	Energy	Capacity
Hopfield (1982)	$\{-1,+1\}^n$	$E(x) = -\frac{1}{2}x^T W x - b^\mathsf{T} x$	$O(n)$
Krotov & Hopfield (2016)	$\{-1,+1\}^n$	$E(x) = -\sum_{i=1}^{n} F((\rho^i)^\mathsf{T} x)$	$\Theta(n^r)$
Demircigil et al. (2017)	$\{-1,+1\}^n$	$E(x) = -\mathrm{LogSumExp}(M^\mathsf{T} x)$	$\Theta(2^{\frac{n}{2}})$
Ramsauer et al. (2020)	$\mathbb{R}^n$	$E(x) = -\mathrm{LogSumExp}(\beta, M^\mathsf{T} x) + \frac{1}{2}x^T x + \beta^{-1}\log d + \max_i \lVert\rho^i\rVert^2/2$	$\Theta(c^{\frac{n-1}{4}})$

Table 2: Transformer-based language models and their reported cross-entropy loss.

Model	Model Size	Data Size	L	Reference
Transformer	1.5B	22B	2.5	Kaplan et al. (2020)
Chinchilla	70B	1.4T	2.2	Hoffmann et al. (2022a)
PaLM 2	16B	100B	2.4	Anil et al. (2023)
GPT-2	8.7B	178B	2.3	Muennighoff et al. (2024)
MiniCPM	2.4B	140B	2.4	Hu et al. (2024c)
Nanotron	1.2B	105B	2.4	Peng et al. (2024)

Table 3: Mean squared error over 1000 iterations between training loss and minimal validation loss for different model configurations and pre-training settings. The last column reports the ratio between N and D^2 for D^* with unit 10^{-10}.

MSE	$D = 167.06$M	$D = 190.38$M	$D^* = 214.18$M	$D = 237.98$M	N/D^{*2} (10^{-10})
$N = 39.95$M	0.07	0.05	**0.04**	0.04	8.71
	$D = 214.18$M	$D = 237.98$M	$D^* = 261.78$M	$D = 285.57$M	N/D^{*2} (10^{-10})
$N = 60.26$M	0.05	0.04	**0.02**	0.02	8.79
	$D = 261.78$M	$D = 285.57$M	$D^* = 309.37$M	$D = 333.17$M	N/D^{*2} (10^{-10})
$N = 80.20$M	0.05	0.04	**0.02**	0.02	8.38

C SOME PROPERTIES OF THE ENERGY FUNCTIONS

We introduce some useful properties of the LogSumExp function defined below. This is particularly useful because The softmax function, widely utilized in the Transformer models, is the gradient of the LogSumExp function. As shown in (Grathwohl et al., 2019), the LogSumExp corresponds to the energy function of the a classifier.

$$\mathrm{LogSumExp}(x) := \log \sum_{i=1}^{n} e^{x_i}, \quad x = (x_1, \ldots, x_n) \in \mathbb{R}^n.$$

Lemma 1. $\mathrm{LogSumExp}(x)$ *is convex.*

Proof.

$$t\mathrm{LogSumExp}(x) + (1-t)\mathrm{LogSumExp}(y) = \log(\sum_{i=1}^{n} e^{x_i})^t (\sum_{i=1}^{n} e^{y_i})^{1-t}$$

$$\geq \log \sum_{i=1}^{n} e^{tx_i + (1-t)y_i} = \mathrm{LogSumExp}(tx + (1-t)y) \quad \forall t \in [0,1].$$

□

Lemma 2. *Suppose* $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, *then we have*

$$\max_{1 \leq i \leq n} x_i < \mathrm{LogSumExp}(x) \leq \max_{1 \leq i \leq n} x_i + \log n.$$

Proof. Taking log on each side of the inequality

$$\exp(\max_{1 \leq i \leq n} x_i) < \sum_{i=1}^{n} \exp(x_i) \leq \sum_{i=1}^{n} \exp(\max_{1 \leq i \leq n} x_i)$$

yields the results. □

Consequently, we have the following smooth approximation for the min function.

Lemma 3. *Suppose* $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, *then we have*

$$\min_{1 \leq i \leq n} x_i - \log n \leq -\mathrm{LogSumExp}(-x) < \min_{1 \leq i \leq n} x_i.$$

Lemma 4. *For* $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \mathbb{R}^n$, *we have*

$$|\mathrm{LogSumExp}(x) - \mathrm{LogSumExp}(y)| \leq \|x - y\|_\infty,$$

where $\|x\|_\infty := \max_{1 \leq i \leq n} |x_i|$.

Proof. Let

$$f(t) := \mathrm{LogSumExp}(tx + (1-t)y), \quad \forall t \in [0,1].$$

According to the mean value theorem, $\exists s \in (0,1)$ such that

$$\mathrm{LogSumExp}(x) - \mathrm{LogSumExp}(y) = f'(s) = \frac{\sum_{i=1}^{n} \exp(sx_i + (1-s)y_i)(x_i - y_i)}{\sum_{i=1}^{n} \exp(sx_i + (1-s)y_i)}.$$

So

$$|\mathrm{LogSumExp}(x) - \mathrm{LogSumExp}(y)| \leq \frac{\sum_{i=1}^{n} \exp(sx_i + (1-s)y_i)\|x - y\|_\infty}{\sum_{i=1}^{n} \exp(sx_i + (1-s)y_i)} = \|x - y\|_\infty.$$

□

C.1 Proof of Proposition 2

Proof. Let $\xi = \max_{1\le i\le d} \|\rho^i\|$, then we have

$$\begin{aligned}2E^{\beta=2}_{\text{MCHN}}(x) &= -\log\left(\sum_{i=1}^{d}\exp(2(\rho^i)^\mathsf{T}x)\right) + \log d + \|x\|^2 + \xi^2\\ &= -\log\left(\sum_{i=1}^{d}\exp(2(\rho^i)^\mathsf{T}x)\right) - \log(\exp(-(\|x\|^2+\xi^2))) + \log d.\end{aligned}$$

So

$$\begin{aligned}2E^{\beta=2}_{\text{MCHN}}(x) - \log d &= -\log(\sum_{i=1}^{d}\exp(2(\rho^i)^\mathsf{T}x - \xi^2 - \|x\|^2))\\ &= -\log(\sum_{i=1}^{d}\exp(\|\rho^i\|^2 - \xi^2 - \|\rho^i - x\|^2)).\end{aligned}$$

Therefore, due to Lemma 4, we have

$$\begin{aligned}|E(x) - (2E^{\beta=2}_{\text{MCHN}}(x) - \log d)| &= |\text{LogSumExp}(\|\rho^i\|^2 - \xi^2 - \|\rho^i - x\|^2) - \text{LogSumExp}(-\|x-\rho^i\|^2)|\\ &\le \max_{1\le i\le d}|\|\rho^i\|^2 - \xi^2| = \max_{1\le i\le d}\|\rho^i\|^2 - \min_{1\le i\le d}\|\rho^i\|^2.\end{aligned}$$

□

D Deferred proofs from Section 5

D.1 Proof of Proposition 4

Proof.

$$\begin{aligned}L(N,D) = H(p_{\widetilde{\mathcal{D}}}, p_\theta) &= -\frac{1}{d'}\sum_{x\in\widetilde{\mathcal{D}}}\log(p_\theta(x)) = -\mathbb{E}_{x\sim p_{\widetilde{\mathcal{D}}}}[\log p_\theta(x)]\\ &= \log Z_\theta \int_{x\in\Omega} P_{\widetilde{\mathcal{D}}}(x)\,\mathrm{d}\mu + \frac{1}{d'}\int_{x\in\Omega}\sum_{i=1}^{d'}\delta(x-\rho^{\sigma(i)})E_{\text{global}}(x)\,\mathrm{d}\mu\\ &= \log Z_\theta + \frac{1}{d'}\sum_{\rho^{\sigma(i)}} E_{\text{global}}(x)\\ &\overset{(a)}{=} \log Z_\theta + \frac{1}{Z_t} - \log l + c \overset{(b)}{\approx} \log Z_t + \frac{1}{Z_t}\end{aligned} \tag{14}$$

where (a) is because $g(\rho^{\sigma(i)}) = 0$, and (b) is due to equation 10, where we have

$$Z_\theta = \int_{x\in\Omega}\exp(-E_{\text{global}}(x))\mathrm{d}x = \frac{l}{e^c}\int_{x\in\Omega}\exp(-E_t(x))\mathrm{d}x, \quad \text{and}$$

$$\log Z_\theta \approx \log l - c + \log\int\exp(-E_t(x))\mathrm{d}x = \log l - c + \log Z_t.$$

□

D.2

Let $B_t(x)$ denote the n-ball with radius t centered at x, $A_{n-1} = \frac{2\pi^{n/2}}{\Gamma(\frac{n}{2})}$ be the hyper-volume of the $(n-1)$-dimensional unit sphere,

$$\gamma(n,r) = \int_0^r t^{n-1}e^{-t}\,\mathrm{d}t, \quad \Gamma(n,r) = \int_r^\infty t^{n-1}e^{-t}\,\mathrm{d}t$$

be the (incomplete) gamma functions. Then

$$\begin{aligned}\int_{x\in B_i} \exp(-\|x-\rho^i\|^2)\,\mathrm{d}x &= \int_{\|x-\rho^i\|<r_i} \exp(-\|x-\rho^i\|^2)\,\mathrm{d}x \\ &= \int_{\|y\|<r_i} \exp(-\|y\|^2)\,\mathrm{d}y \\ &= \int_0^{r_i}\int_{\partial B_t(0)} e^{-t^2}\,\mathrm{d}\mathcal{H}^{n-1}\mathrm{d}t \\ &= \int_0^{r_i} e^{-t^2}\mathcal{H}^{n-1}(\partial B_t)\mathrm{d}t \\ &= \int_0^{r^i} e^{-t^2}A_{n-1}t^{n-1}\,\mathrm{d}t \\ &= \frac{2\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2})}\int_0^{r_i} t^{n-1}e^{-t^2}\,\mathrm{d}t \\ &= 2\pi^{\frac{n}{2}}\frac{\gamma(n,r_i)}{\Gamma(\frac{n}{2})}\end{aligned}$$

Lemma 5. *The incomplete gamma function $\gamma(n,r)$ satisfies*

$$e^{-r}\frac{r^n}{n} \le \gamma(n,r) \le \frac{r^n}{n}$$

Proof. For $0 \le x \le r$, we have

$$x^{n-1}e^{-r} \le x^{n-1}e^{-x} \le x^{n-1}.$$

Integrating from 0 to r on each side yields the result. □

Theorem D.1 (Stirling's approximation)**.** *For any complex z, the Stirling's approximation gives that*

$$\Gamma(z) = \sqrt{\frac{2\pi}{z}}\left(\frac{z}{e}\right)^z\left(1+O(\frac{1}{z})\right).$$

For large z,

$$\Gamma(z+1) \approx \sqrt{2\pi z}\left(\frac{z}{e}\right)^z.$$

E TRANSFORMER DETAILS: USING GPT-2 AS AN EXAMPLE

The original GPT-2 model was trained on a 40GB large dataset called WebText that is made of data derived from outbound links from Reddit. The model is trained on the next sentence prediction (NSP) task in a self-supervised manner. A pre-trained tokenizer can be applied to convert the text into tokens using a fixed vocabulary. A max token length $T_{\max}$ (e.g., $T_{\max} = 1024$) is set, so during training, if the number of tokens is greater than $T_{\max}$, the documents will be truncated. The model is trained causally, which means that the prediction for the next token only depends on the inputs from earlier tokens. The model was trained with a global batch size of 512, and the test perplexity still improves if given more training time.

GPT-2 uses a byte-level version of Byte Pair Encoding (BPE), and the vocabulary size is $n_{\text{voc}} = 50,257$. The hidden dimension for the medium size model is $d_{\text{emb}} = 1024$. So the input sequence is of $T_{\max}d_{\text{emb}}$ dimension. These sequences are passed through the model. For the medium size model, these include 24 transformer encoder blocks with 1024 hidden units and 16 self-attention heads (i.e., $l = 24, d_{\text{emb}} = 1024, n_h = 16$). The number of parameters used for word embedding is $n_{\text{voc}}\cdot d_{\text{emb}}$. The number of parameters in the multi-head attention layer is $l\cdot n_h\cdot(3\cdot d_{\text{emb}}\cdot d_{\text{emb}}/n_h) = 3ld_{\text{emb}}^2 = 75,497,472$. The number of parameters in the dense weights and layer normalization is $l(d_{\text{emb}}^2+2d_{\text{emb}}) = ld_{\text{emb}}^2+2ld_{\text{emb}} = 25,214,976$, and the number of parameters in the feed-forward weight matrices and bias is $l(2d_{\text{emb}}\cdot d_{\text{FF}} + d_{\text{emb}} + d_{\text{FF}}) = 6ld_{\text{emb}}^2 + 4ld_{\text{emb}} = 151,093,248$, with

$d_{\text{FF}} = 3072 = 3d_{\text{emb}}$. As we can observe, the multi-head attention and feed-forward layers account for most of the parameters in the model, and $N \approx Ald_{\text{emb}}^2$ with some constant $A \approx 10$ in this case.

The loss used for the GPT-2 model is the log-probability of a dataset divided by the number of canonical units (e.g., a character, a byte, a word), which is equivalent to the cross-entropy loss. The cross-entropy loss is commonly used to measure the divergence between the predicted probabilities and the true labels. For the NSP task, the model is trained to predict the next token in a sequence based on the context of the previous tokens. So the cross-entropy is taken between the predicted probabilities $\Pr_\theta(x_i)$ of the token x_i and the labels' probabilities $\Pr_D(x_i)$ for all tokens x_i in the vocabulary, i.e.,

$$-\frac{1}{D}\sum_{i=1}^{D}\log(p_\theta(x_i)) = -\sum_{x\sim p_D} p_D(x)\log(p_\theta(x)).$$

Another commonly used loss is the perplexity, which is equivalent to the exponentiated version of the cross-entropy.

F Experimental Details

F.1 Configurations

We follow the OpenELM (Mehta et al., 2024) architecture and choose the following configurations such that the number of transformer parameters are about 40M, 60M, and 80M.

Parameter	**Model 1**	**Model 2**	**Model 3**
Number of Transformer Parameters	39.95M	60.26M	80.20M
Model Dimension	954	1280	1440
Number of Transformer Layers	8	8	8
Number of KV Heads	3, 3, 3, 3, 4, 4, 4, 5	3, 3, 3, 3, 4, 4, 4, 5	3, 3, 3, 4, 4, 4, 5, 5
Number of Query Heads	6, 6, 6, 6, 8, 8, 8, 10	6, 6, 6, 6, 8, 8, 8, 10	6, 6, 6, 8, 8, 8, 10, 10

Hyperparameters for pre-training are listed below.

Parameter	**Detail**
Tokens per iteration	491,520
Vocabulary Size	50,257
Activation Function	swish
Attention Dropout	0.1
Embedding Dropout	0.1
Head Dimension	64
Initializer Range	0.02
Max Context Length	1024
Normalization Layer	rms_norm
Normalize QK Projections	True
QKV Multipliers	0.5, 1.0
Batch size	12
AdamW	$\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 10^{-8}$

F.2 Robustness to Deduplication

In order to further confirm the validity of our analyses, we conduct five independent training runs of Model 1 to assess the model's robustness and the consistency of its performance across different training instances. Fig. 3 illustrates the CE loss over the course of training iterations for each run, along with the minimum validation loss achieved during each run. It can be observed that the training runs exhibit varying degrees of performance, as indicated by the different trajectories of the CE loss curves. The minimum validation loss remains consistent to the second decimal place across various training runs, and is achieved at approximately the 6000th training step.

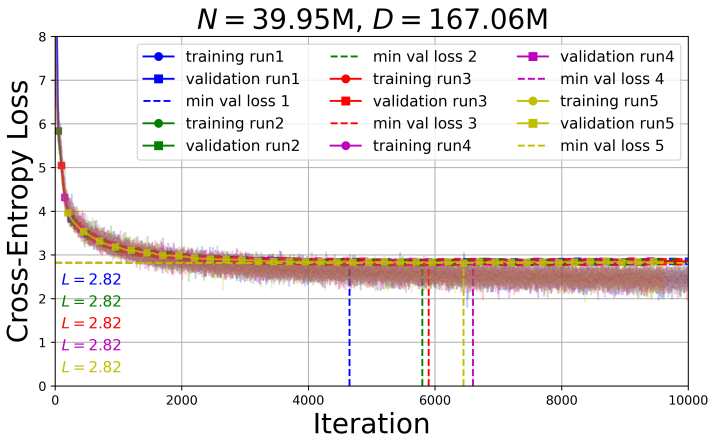


Figure 3: The cross-entropy loss for one model configuration during pre-training (depicted with dots) and validation (depicted with squares) across five separate training runs. The minimal attainable validation loss is represented by dashed lines. Each individual run's performance is distinguished by a unique color, and the y-axis highlights the lowest validation loss for each respective run.

F.3 Additional Results

Figured below are the training dynamics of the models in Table 3 for visual inspection.

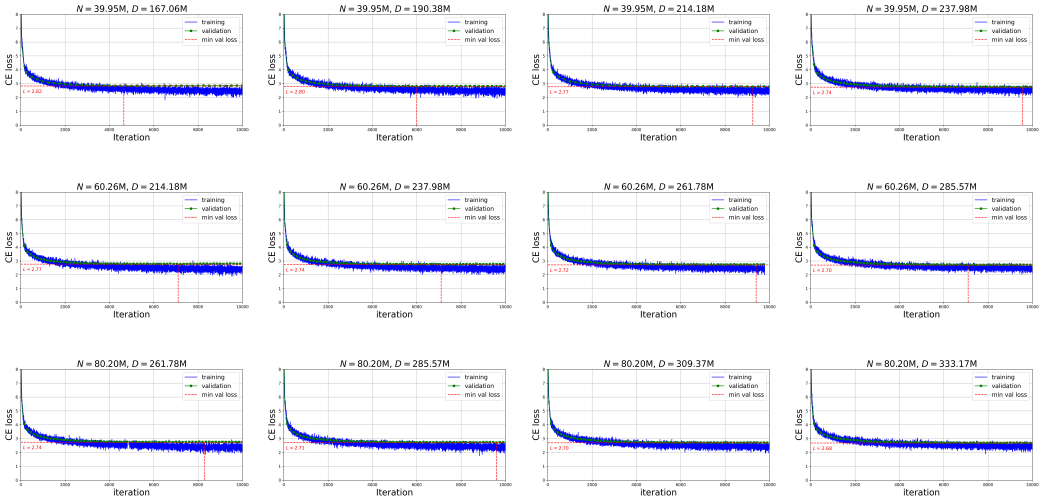


Figure 4: Cross-entropy losses of eight models employing the OpenELM architecture as presented in Table 3.