From Entropy Rate to Redundancy: Information Dynamics in Large Language Models

Jessica E. Liang
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
jeliang@seas.upenn.edu

Abstract

Large language models (LLMs) achieve impressive performance, yet the mechanisms by which information flows and adapts during fine-tuning remain underexplored. We introduce *entropy rate* as a dynamic, space–time information-theoretic metric that captures how uncertainty propagates across layers and evolves across epochs. Building on this foundation, we derive the *redundancy score*, a tractable approximation that quantifies the predictability of each layer's representations from its neighbors. Layers with high redundancy contribute little novel information and are strong candidates for structured pruning. Empirical studies on RoBERTa-base (GLUE benchmark) show that redundancy-score pruning achieves substantial compression while preserving accuracy, outperforming knowledge-entropy pruning, LayerDrop, and SlimLLM. Beyond compression, redundancy profiles reveal consistent architectural patterns, with mid-layer peaks corresponding to dynamic representational activity. These findings position entropy rate and redundancy score as principled, interpretable tools for analyzing, optimizing, and compressing foundation models in natural language understanding and reasoning.

1 Introduction

Large language models (LLMs), such as RoBERTa, GPT, and T5, have achieved remarkable success across a wide spectrum of natural language understanding and generation tasks. Despite their empirical effectiveness, the internal information dynamics that underpin their learning and adaptation remain incompletely understood. In particular, questions persist regarding how uncertainty, redundancy, and information flow evolve within these deep architectures during fine-tuning, and how such dynamics relate to functional specialization, knowledge acquisition, and efficient model compression.

Previous research has sought to probe LLM representations using a range of tools, including representational similarity analysis [25, 18], probing classifiers [2, 3], and information-theoretic measures such as mutual information (MI) [29, 27, 12]. While these approaches have revealed important structural and geometric properties of deep neural representations, they are typically limited to static, layerwise snapshots. As a result, they offer only partial visibility into the temporal evolution and dynamic propagation of information across layers and epochs.

A recent line of work has introduced knowledge entropy as a means to quantify uncertainty in LLM outputs and internal memory utilization [17]. However, knowledge entropy is inherently static, capturing only the aggregate uncertainty in a single layer or at a single epoch. This motivates the need for dynamic, process-level metrics that can more fully characterize the propagation and transformation of information as learning unfolds.

In this paper, we propose *entropy rate* as a principled, dynamic information-theoretic metric for analyzing LLM fine-tuning. Entropy rate measures the conditional uncertainty in a layer's represen-

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: SPIGM.

tations at a given epoch, conditioned on preceding layers and previous epochs, thereby capturing both spatial (layerwise) and temporal (epochwise) information flow. By formulating representation dynamics as a space–time Markov chain, we derive efficient approximations of entropy rate based on Gaussianization and cosine similarity of finite differences. From this framework, we introduce the *redundancy score*, a monotone reparameterization of entropy rate that quantifies how predictable a layer's representations are from its neighbors. Layers with high redundancy scores contribute little novel, task-relevant information and are natural candidates for structured pruning.

We empirically evaluate redundancy score and entropy rate on $\text{RoBERTa}_{\text{base}}$ across eight GLUE benchmark tasks, comparing against knowledge entropy, SlimLLM, and random structured dropout. Our results reveal a universal, non-monotonic redundancy profile—low at the input/output layers and peaking in the mid-layers—indicating zones of heightened representational flexibility. Pruning layers with the highest redundancy score preserves accuracy while significantly reducing model complexity, outperforming knowledge-entropy pruning, SlimLLM, and LayerDrop.

Contributions. Our main contributions are:

- We introduce entropy rate as a dynamic, space–time information-theoretic metric for characterizing knowledge propagation and uncertainty dynamics in LLM fine-tuning.
- We derive the *redundancy score* from entropy rate, providing a tractable and interpretable criterion for identifying prunable layers.
- Through extensive experiments on GLUE, we show that redundancy score uncovers robust, universal specialization patterns in LLMs and enables effective structured pruning with minimal performance loss.
- We clarify the distinction between entropy rate and redundancy score highlighting their complementary roles in understanding model dynamics.

Overall, our findings establish entropy rate and redundancy score as principled, interpretable, and practical tools for analyzing, diagnosing, and compressing large language models, enabling more efficient and robust neural NLP systems.

2 Related Work

Understanding the internal information dynamics of LLMs during training and fine-tuning has been an active and rapidly developing area of research. Early investigations primarily focused on static analyses using probing [2, 3], representational similarity [25, 18], and information-theoretic measures such as mutual information (MI) [29, 27, 30, 12, 10], to provide snapshots of knowledge, feature geometry, or representational structure in pretrained models. However, these approaches generally do not capture the dynamic, process-level evolution of internal representations that occurs throughout fine-tuning.

Knowledge Entropy and Representational Dynamics. The concept of *knowledge entropy*, introduced by Kim et al. [17], provides an information-theoretic view on the evolution of output entropy during LLM pretraining and its link to knowledge acquisition capacity. They show that declining output entropy signals a reduction in learning ability, but their framework focuses on output distributions and single-layer metrics. Our approach extends this line by adopting a multi-layer, space-time formulation of *entropy rate*, capturing how uncertainty and information propagate not only through depth but also across epochs. In parallel, Skean et al. [28] highlight the importance of intermediate layers, demonstrating via information-theoretic and geometric analysis that these layers often yield the richest representations for downstream tasks.

Mutual Information in Neural Networks. The Information Bottleneck (IB) principle [29] and its deep learning applications [27, 12, 10] have inspired a substantial body of work using MI to understand learning and generalization. Tishby et al. [30] argue that deep networks can be characterized by MI between layers and input/output variables, which relate to theoretical generalization bounds. Extensions such as those by Goldfeld et al. [10, 11] address practical MI estimation in deep architectures. While MI-based metrics track important spatial dependencies, their use in high-dimensional settings is often limited by computational challenges and statistical noise [24]. Furthermore, most MI analyses provide only static or per-layer insight, neglecting the temporal dynamics that unfold

during fine-tuning. Our entropy rate framework circumvents these limitations by providing a tractable, theoretically grounded measure of temporal and spatial information propagation.

Entropy and Uncertainty in Language Models. A rapidly expanding line of research leverages entropy-based metrics to probe LLM uncertainty, robustness, and generalization. The so-called "entropy law" connecting compression to performance has been explored in [36], while panoptic analyses link compression and cross-entropy to scaling laws [23]. Entropy minimization has emerged as an effective regularizer in both training and inference [1, 4], and high-entropy tokens are linked to model uncertainty and robustness [34]. More nuanced metrics, such as kernel language entropy [22] and semantic entropy [9], enable context-sensitive evaluation. Recent studies have also explored entropy in the context of memory compression [6], privacy [13], dataset curation [33], memorization [16], and test-time scaling [33]. Yet, most entropy analyses remain static or focus on token-level uncertainty, failing to capture dynamic information flow over time and depth.

Entropy Rate and Dynamic Information Flow. Our work is distinctive in introducing *entropy rate* as a space-time metric, tracking how uncertainty propagates through both layers and epochs during LLM fine-tuning. By moving beyond static or token-level analyses, entropy rate provides a fundamentally new perspective on representational evolution, saturation, and the flow of knowledge. Related efforts on entropic distribution matching [19] and adaptive regularization [35] hint at the broader importance of dynamic, process-level information metrics for model adaptation and optimization.

Other Related Approaches. Cosine similarity and Centered Kernel Alignment (CKA) [25, 18] are commonly used to compare layer representations but lack grounding in information theory and do not quantify learning capacity or knowledge saturation. Recent work has also used entropy-based metrics in continual learning for detecting knowledge plateaus or catastrophic forgetting [5], but such techniques are rarely applied to standard LLM fine-tuning scenarios. Our entropy rate formulation provides a unified and practical tool for diagnosing knowledge saturation and representational bottlenecks in both standard and continual learning regimes.

Summary. While prior research has built a foundation for probing, quantifying, and analyzing LLM representations, our contribution lies in advancing a dynamic, theoretically principled, and computationally practical entropy rate metric. This framework allows us to diagnose knowledge acquisition, representational redundancy, and saturation throughout both space (layers) and time (epochs), bridging the gap between information theory and the evolving internal dynamics of large language models.

3 Proposed Method

3.1 Markov Chain Structure for Representations in the Space-Time Domain

During LLM fine-tuning, model representations evolve both hierarchically across layers and progressively across epochs. Let X_l^t denote the hidden representation (or activation) at layer l and epoch t. Typically, X_l^t is a high-dimensional vector (or tensor, for batched inputs) capturing the intermediate output of the l-th transformer block at epoch t. These representations encode rich semantic and syntactic features, which change both with model depth and as fine-tuning proceeds.

The evolution of these representations can be naturally modeled as Markov chains, since each state primarily depends on its immediate predecessor. In the spatial (layerwise) domain, the representation at each layer depends chiefly on the output of the previous layer, while in the temporal (epochwise) domain, the representation at each epoch is mainly determined by the prior epoch due to incremental parameter updates.

Formally, the Markov dependencies can be expressed as

$$X_1 \to X_2 \to \cdots \to X_L, \tag{1}$$

in the spatial (layer) domain, and as

$$X^1 \to X^2 \to \cdots \to X^T, \tag{2}$$

in the temporal (epoch) domain. Here, L denotes the number of layers and T the number of epochs.

These Markov chain structures justify the use of dynamic, process-level information-theoretic metrics, the *entropy rate*, to characterize how uncertainty, information, and knowledge propagate through space (layers) and time (epochs) during LLM fine-tuning.

3.2 Entropy Rate in LLM Fine Tuning

The entropy of a random variable measures its uncertainty [7]. The entropy rate quantifies the uncertainty or variability of the representations generated by each layer across epochs, conditioned on its previous state and neighboring layers. Formally, we define entropy rate of LLM fine tuning in space-time as:

$$H(X_{l+1}^{t+1}|X_l^{t+1}, X_{l-1}^{t+1}, \cdots, X_1^{t+1}, X_{l+1}^t, X_{l+1}^{t-1}, \cdots, X_{l+1}^1) \tag{3}$$

where X_l^t denotes the hidden representation (i.e., the output embedding or activation) of layer l at epoch t. This quantity captures the conditional uncertainty in the representation produced by layer $l+1$ at epoch $t+1$, given all lower-layer representations at the current epoch and all past representations of layer $l+1$.

Based on the Markov chains in (1) and (2), (3) can be simplified as

$$H(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t) = -\sum_x P(x_{l+1}^{t+1}, x_l^{t+1}, x_{l+1}^t) \log \frac{P(x_{l+1}^{t+1}, x_l^{t+1}, x_{l+1}^t)}{P(x_l^{t+1}, x_{l+1}^t)} \tag{4}$$

The Markovian property justifies the definition and computation of entropy rate in our proposed space-time analysis framework, providing a structured and rigorous means of quantifying knowledge dynamics.

We can explicitly write the representation X_{l+1}^{t+1} as a function of the current input, multi-head attention weights, and feed-forward weights [31]:

$$X_{l+1}^{t+1} = f_{l+1}\left(\mathrm{MHA}_{l+1}(X_l^{t+1}; W_{l+1}^{\mathrm{attn}}),\ W_{l+1}^{\mathrm{ffn}}\right) \tag{5}$$

$$\mathrm{MHA}_{l+1}(X) = \mathrm{Concat}\left(\mathrm{head}_1, \ldots, \mathrm{head}_h\right) W_{l+1}^O \tag{6}$$

$$\mathrm{head}_i = \mathrm{Attention}(XW_{l+1,i}^Q,\ XW_{l+1,i}^K,\ XW_{l+1,i}^V) \tag{7}$$

where W_{l+1}^{attn} and W_{l+1}^{ffn} denote the set of attention and feed-forward weights in layer $l+1$.

Note that X_{l+1}^{t+1} depends not only on the lower-layer representations at the current epoch, but also on its own previous state X_{l+1}^t via the parameter update process:

$$X_{l+1}^{t+1} = f_{l+1}(X_l^{t+1};\ \theta_{l+1}^{t+1}), \tag{8}$$

where θ_{l+1}^{t+1} is obtained from θ_{l+1}^t by a gradient update that uses X_{l+1}^t (or gradients computed from it). Thus, X_{l+1}^{t+1} is indirectly dependent on X_{l+1}^t, reflecting the memory effect of weight updates over epochs.

In the analysis of information dynamics within LLMs, directly computing quantities like entropy rate is often intractable due to the high dimensionality and complex, nonlinear dependencies between layers and time steps. These information-theoretic measures, however, are critical for understanding how knowledge is propagated, retained, or transformed across the network's depth (layers) and breadth (time or input sequence). Entropy rate quantifies the uncertainty or information content introduced at each layer over time. Approximating these quantities using tractable proxies such as first-order Gaussian approximations [10][26] or cosine similarity between finite differences of embeddings—allows us to efficiently and insightfully characterize the model's internal information flow. These approximations provide practical tools for diagnosing bottlenecks, identifying redundancy, and informing pruning or compression strategies in modern transformer-based architectures. In this paper, we propose approximation methods for entropy rate.

3.3 Entropy Rate Approximation

Let J_l denote the Jacobian of layer l with respect to a local input perturbation around the state (X_l^t, X_{l+1}^t). Formally, it is defined as the first-order derivative of the layer's output at time $t+1$ with respect to the local input vector x:

$$J_l = \left.\frac{\partial X_l^{t+1}}{\partial x}\right|_{x=(X_l^t, X_{l+1}^t)}, \tag{9}$$

and analogously for layer $l+1$,

$$J_{l+1} = \left.\frac{\partial X_{l+1}^{t+1}}{\partial x}\right|_{x=(X_l^t, X_{l+1}^t)}. \tag{10}$$

Here, x represents a perturbation vector sampled from a local Gaussian distribution centered at the current state. Although X_l^{t+1} and X_{l+1}^{t+1} are evaluated at time $t+1$, the Jacobians are computed at time t to capture the local linear behavior of the forward dynamics.

Define $u = X_{l+1}^{t+1}$ and $v = (X_l^{t+1}, X_{l+1}^t)$. Under a first-order Taylor expansion around $x_0 = (X_l^t, X_{l+1}^t)$, we have:

$$u \approx u_0 + J_{l+1}\,\delta x, \tag{11}$$

$$v \approx v_0 + \begin{pmatrix} J_l \\ J_{l+1} \end{pmatrix} \delta x, \tag{12}$$

where $\delta x \sim \mathcal{N}(0, \sigma^2 I)$ is a small perturbation, and the nominal (unperturbed) values are:

$$u_0 = X_{l+1}^{t+1}\big|_{x=x_0}, \tag{13}$$

$$v_0 = \big(X_l^{t+1}, X_{l+1}^t\big)\,\big|_{x=x_0}. \tag{14}$$

The constants u_0 and v_0 represent the outputs at the expansion point and merely shift the Gaussian mean; they do not affect the covariance and thus do not contribute to the entropy computation.

Theorem 1 (First-Order Entropy Rate Approximation)**.** *Under a first-order linear–Gaussian approximation around $x = (X_l^t, X_{l+1}^t)$ with perturbation $\delta x \sim \mathcal{N}(0, \sigma^2 I)$, the entropy rate of layer $l+1$ admits the expansion*

$$H\big(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t\big) \;\approx\; \frac{d_{l+1}}{2}\,\ln\!\big(2\pi e\,\sigma^2\big) \;-\; \frac{1}{2}\,\mathrm{tr}(A), \tag{15}$$

where

$$A \;=\; (\Sigma_{uu})^{-1/2}\,\Sigma_{uv}\,\Sigma_{vv}^{-1}\,\Sigma_{vu}\,(\Sigma_{uu})^{-1/2}, \tag{16}$$

and the local covariance blocks are

$$\Sigma_{uu} \;=\; \sigma^2 J_{l+1} J_{l+1}^\top, \tag{17}$$

$$\Sigma_{vv} \;=\; \sigma^2 \begin{pmatrix} J_l J_l^\top & J_l J_{l+1}^\top \\ J_{l+1} J_l^\top & J_{l+1} J_{l+1}^\top \end{pmatrix}, \tag{18}$$

$$\Sigma_{uv} \;=\; \sigma^2\, J_{l+1} \begin{pmatrix} J_l \\ J_{l+1} \end{pmatrix}^\top. \tag{19}$$

The proof of this Theorem is provided in Appendix A.

Theorem 2 (Cosine-Similarity Approximation of Entropy Rate)**.** *Under the first-order linear–Gaussian and finite-difference approximations, the entropy rate*

$$H\big(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t\big)$$

can be estimated by

$$H\big(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t\big) \approx \frac{d_{l+1}}{2}\ln\!\big(2\pi e\,\sigma^2\big) \;-\; \frac{1}{2(B-1)}\sum_{i=1}^{B-1}\cos^2\!\big(\Delta\mathbf{z}_{l+1,i},\,\Delta\mathbf{z}_{l,i}\big), \tag{20}$$

where

$$\begin{aligned} \Delta\mathbf{z}_{l,i} &\approx \mathbf{z}_{l,i+1} - \mathbf{z}_{l,i}, \\ \Delta\mathbf{z}_{l+1,i} &\approx \mathbf{z}_{l+1,i+1} - \mathbf{z}_{l+1,i}, \end{aligned} \tag{21}$$

and $\mathbf{z}_{l,i}$ and $\mathbf{z}_{l+1,i}$ denote the embeddings at layer l and $l+1$ on the i-th sample of a batch of size B.

The proof of this Theorem is provided in Appendix B.

To obtain the conditional entropy estimate over a full epoch, we compute the average of the batch-level estimates. Suppose the epoch consists of M batches, each of size B. For the m-th batch, the estimate is given by:

$$H_{l+1}^{(m)} \approx \frac{d_{l+1}}{2}\ln\big(2\pi e\,\sigma^2\big) - \frac{1}{2(B-1)}\sum_{i=1}^{B-1}\cos^2\big(\Delta\mathbf{z}_{l+1,i}^{(m)},\,\Delta\mathbf{z}_{l,i}^{(m)}\big). \tag{22}$$

Then, the epoch-level conditional entropy is obtained by averaging over all M batches:

$$\overline{H}_{l+1} = \frac{1}{M}\sum_{m=1}^{M} H_{l+1}^{(m)} \tag{23}$$

$$= \frac{d_{l+1}}{2}\ln\big(2\pi e\,\sigma^2\big) - \frac{1}{2M(B-1)}\sum_{m=1}^{M}\sum_{i=1}^{B-1}\cos^2\big(\Delta\mathbf{z}_{l+1,i}^{(m)},\,\Delta\mathbf{z}_{l,i}^{(m)}\big). \tag{24}$$

This expression provides a stable estimate of the conditional entropy for layer $l+1$ across the full training epoch.

In Appendix C, we provide the computational complexity, memory and storage requirements for the exact computation, Theorems 1 and 2. For representation dimension d and batch size B, the computational complexity for Theorem 1 is $O(d^3 + d^2B)$, and for Theorem 2 is $O(dB)$.

Subsequently, we define *Redundancy Score* R_{l+1}:

$$R_{l+1} := \frac{1}{2M(B-1)}\sum_{m,i}\cos^2\big(\Delta\mathbf{z}_{l+1,i}^{(m)},\,\Delta\mathbf{z}_{l,i}^{(m)}\big) \tag{25}$$

The details are described in Appendix D. We will therefore refer to R_{l+1} as the *Redundancy Score* (higher values indicate greater redundancy and, thus, a more prunable layer).

4 Experimental Results

We evaluate the entropy rate during both full fine-tuning and Low Rank Adaptation (LoRA) [15] using RoBERTa [20] on the GLUE [32] benchmark. In addition, we compare entropy rate with knowledge entropy during full fine-tuning, and further apply entropy rate as a criterion for layer pruning. All experiments are conducted in a Google Colab environment equipped with an NVIDIA A100 GPU, ensuring a consistent computational platform for entropy rate analysis and layer pruning of RoBERTa on standard natural language understanding tasks. The runtime for the experiments was approximately 3 hours.

4.1 Entropy Rate in Full Fine Tuning

In Fig. 1, we summarize the entropy rate (redundancy score) versus the layer index (starting layer 2 to compute redundancy score) in $\text{RoBERTa}_{\text{base}}$ fine tuning for eight datasets on the GLUE benchmark. Across all datasets, the redundancy score as a function of layer depth exhibits a highly consistent and distinctive profile. Specifically, the redundancy score remains low in the input and output layers, but increases sharply through the early and middle layers, reaching a pronounced peak in the upper-middle layers of the model (typically layers 3 to 8). This non-monotonic "$\cap$-shaped" pattern is robust to both the choice of dataset and the epoch of fine-tuning. The peak redundancy score values, observed in the middle layers, often surpass 0.2–0.3, while the redundancy score in the final layer can be as low as 10^{-3}. Minor variations in the exact location and amplitude of the redundancy score peak are observed across different datasets, reflecting some degree of task specificity, yet the qualitative trend remains universal. Furthermore, this profile is stable across training epochs, with early, middle, and late layers maintaining their respective redundancy score characteristics throughout fine-tuning.

These results indicate a clear functional specialization across the layers of large language models. The low redundancy score in the input layers suggests stable feature extraction, while the elevated redundancy score in the middle layers marks a zone of increased representational uncertainty and flexibility, likely corresponding to dynamic knowledge integration and task adaptation. Output layers,

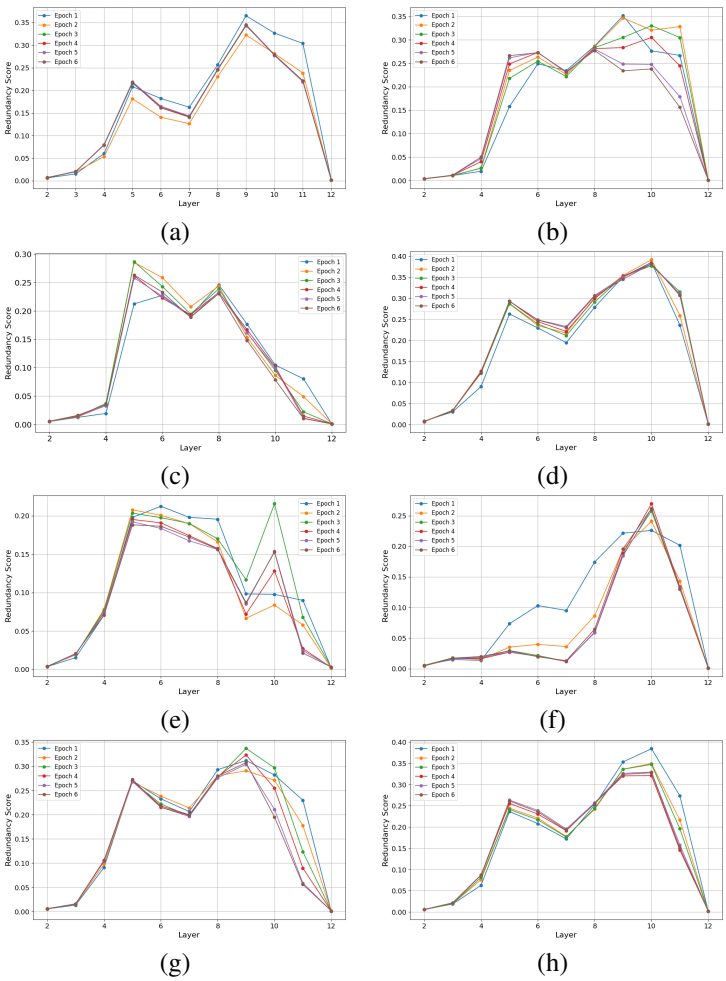

Figure 1: The redundancy score versus layer index in RoBERTa$_{\text{base}}$ with full fine tuning for 6 epochs using eight datasets on the GLUE benchmark, (a) MNLI, (b) MRPC, (c) SST-2, (d) CoLA, (e) QNLI, (f) QQP, (g) RTE, (h) STS-B.

in contrast, return to low redundancy score, reflecting highly deterministic, task-aligned decoding. The universality of this $\cap$-shaped redundancy score profile implies that redundancy score is an intrinsic property of transformer architectures during fine-tuning, largely independent of specific dataset or task.

Consequently, redundancy score can serve as a principled metric for identifying which layers are most amenable to pruning or compression. Layers with high redundancy score exhibit greater redundancy in their representations, often reflecting dynamic adaptation rather than essential information. As a result, pruning these layers with higher redundancy score is less likely to harm the model's predictive accuracy or core capabilities, since their information is either noisy or can be compensated by neighboring layers. In contrast, low-redundancy-score layers encode more stable and indispensable representations. This observation justifies a redundancy-score-guided pruning strategy, where layers with the highest redundancy score are removed to achieve efficient model compression without significant degradation in performance.

4.2 Application to LLM Pruning

We prune layers using the *redundancy score* R_l from (25), a positive, monotone reparameterization of the layerwise entropy-rate estimator, rather than the raw entropy rate. Intuitively, R_l quantifies how much a layer's representation change is predictable from (or duplicative of) neighboring layers, i.e.,

how little novel, task-relevant information it contributes. Layers with higher R_l thus encode more redundant or noisy variation and are stronger pruning candidates. We therefore rank layers by R_l (higher is more prunable) and remove those at the top of the ranking subject to architectural constraints (e.g., preserving embeddings and the output head). This targeted criterion preferentially eliminates duplicative computation, reducing model size and inference latency while maintaining—often after a brief recovery fine-tune—accuracy and generalization.

Table 1 reports pruning results on RoBERTa$_{\text{base}}$ across the GLUE benchmark (MNLI, MRPC, SST-2, CoLA, QNLI, QQP, RTE, STS-B). The first row gives the fully fine-tuned baseline after 6 epochs, using the task-appropriate metric (accuracy for MNLI/QNLI/RTE/SST-2, Matthews correlation for CoLA, F1/accuracy for MRPC/QQP, and Pearson correlation for STS-B). For pruning, we remove four of the twelve transformer layers using four criteria: (1) pruning the four layers with the highest *redundancy score* R_l (a positive, monotone reparameterization of our layerwise entropy-rate estimator), (2) pruning the four layers with the highest knowledge entropy (KE), (3) random structured LayerDrop following Fan et al. [8], and (4) SlimLLM layer pruning [14]. After each pruning/dropout operation, we fine-tune for 5 additional epochs. The post-pruning results are summarized in rows 2–5.

Table 1: GLUE benchmark accuracy (or correlation for STS-B) for RoBERTa$_{\text{base}}$ under full fine-tuning, entropy rate pruning, knowledge entropy pruning, and random layer dropout.

Method	**MNLI**	**MRPC**	**SST-2**	**CoLA**	**QNLI**	**QQP**	**RTE**	**STS-B**
No Pruning	0.7840	**0.8480**	**0.9220**	**0.8274**	**0.8731**	**0.8386**	**0.7509**	**0.9075**
Our ER Pruning	**0.7860**	**0.8480**	0.9186	0.8255	0.8548	0.8361	0.7437	0.9063
KE Pruning [17]	0.3668	0.6838	0.7294	0.6913	0.5054	0.6318	0.5271	0.3125
Layer Dropout [8]	0.6910	0.8235	0.9186	0.6913	0.8371	0.8003	0.7256	0.8938
SlimLLM [14]	0.7660	0.8088	0.8979	0.8092	0.8548	0.8359	0.6787	0.9043

Our results show that redundancy-score (ER-derived) pruning closely tracks the full fine-tuning baseline, indicating that R_l is effective at identifying layers that contribute little novel, task-relevant information. In contrast, KE-based pruning causes substantial degradation across all tasks, suggesting it is not a reliable signal for structured layer removal in RoBERTa. LayerDrop yields intermediate performance—better than KE but consistently below the redundancy-score criterion, with especially large gaps on linguistically challenging tasks such as CoLA and QNLI. SlimLLM is competitive but generally trails our method and the no-pruning baseline, with parity on QNLI and near-parity on QQP/STS-B. Overall, these findings support the redundancy score as a robust and principled pruning signal, enabling meaningful compression with minimal loss in downstream performance.

5 Conclusions and Future Work

We introduced a space–time framework for analyzing information dynamics during LLM fine-tuning by modeling hidden representations as Markov chains across depth (layers) and time (epochs). Leveraging this structure, we derived tractable approximations to entropy rate via a first-order linear–Gaussian analysis and a cosine-similarity surrogate and proposed the *Redundancy Score* R_ℓ as a positive, monotone reparameterization that quantifies how predictable a layer's representation change is from its neighbors. Empirically, R_ℓ provides an effective signal for structured layer pruning: on RoBERTa$_{\text{base}}$ across GLUE (Table 1), redundancy-score pruning closely preserves the full fine-tuning baseline and outperforms knowledge-entropy pruning and LayerDrop, while remaining competitive with SlimLLM.

Several promising directions emerge from our study. First, extending entropy rate analysis to even larger LLMs (e.g., GPT) and multi-modal architectures may further illuminate universal principles of deep learning dynamics. Second, combining entropy rate with other dynamic metrics, such as mutual information rate or Fisher information, could yield a richer, multi-faceted understanding of knowledge acquisition, retention, and forgetting. Third, exploring entropy-rate-guided pruning in continual, multi-task, or low-resource settings may unlock new avenues for efficient adaptation and deployment of foundation models.

References

[1] Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*, 2025.

[2] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *ICLR*, 2017.

[3] Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What do neural machine translation models learn about morphology? In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[4] Haw-Shiuan Chang, Nanyun Peng, Mohit Bansal, Anil Ramakrishna, and Tagyoung Chung. Real sampling: Boosting factuality and diversity of open-ended generation via asymptotic entropy. *arXiv preprint arXiv:2406.07735*, 2024.

[5] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–547, 2018.

[6] Feng Cheng, Cong Guo, Chiyue Wei, Junyao Zhang, Changchun Zhou, Edward Hanson, Jiaqi Zhang, Xiaoxiao Liu, Hai Li, and Yiran Chen. Ecco: Improving memory bandwidth and capacity for llms via entropy-aware cache compression. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, pages 793–807, 2025.

[7] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2006.

[8] Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. *ICLR*, 2020.

[9] Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, et al. Efficiently scaling llm reasoning with certaindex. *arXiv preprint arXiv:2412.20993*, 2024.

[10] Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating information flow in deep neural networks. *ICML*, 2019.

[11] Ziv Goldfeld, Kristjan Greenewald, Jonathan Weed, and Yury Polyanskiy. Optimality of the plug-in estimator for differential entropy estimation under gaussian convolutions. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 892–896. IEEE, 2019.

[12] Ziv Goldfeld and Yury Polyanskiy. The information bottleneck problem and its applications in machine learning. *IEEE Journal on Selected Areas in Information Theory*, 1(1):19–38, 2020.

[13] Tianle Gu, Zongqi Wang, Kexin Huang, Yuanqi Yao, Xiangliang Zhang, Yujiu Yang, and Xiuying Chen. Invisible entropy: Towards safe and efficient low-entropy llm watermarking. *arXiv preprint arXiv:2505.14112*, 2025.

[14] Jialong Guo, Xinghao Chen, Yehui Tang, and Yunhe Wang. Slimllm: Accurate structured pruning for large language models. *ICML*, 2025.

[15] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[16] Yizhan Huang, Zhe Yang, Meifang Chen, Jianping Zhang, and Michael R Lyu. Entropy-memorization law: Evaluating memorization difficulty of data in llms. *arXiv preprint arXiv:2507.06056*, 2025.

[17] Jiyeon Kim, Hyunji Lee, Hyowon Cho, Joel Jang, Hyeonbin Hwang, Seungpil Won, Youbin Ahn, Dohaeng Lee, and Minjoon Seo. Knowledge entropy decay during language model pretraining hinders new knowledge acquisition. *ICLR*, 2025.

[18] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[19] Ziniu Li, Congliang Chen, Tian Xu, Zeyu Qin, Jiancong Xiao, Ruoyu Sun, and Zhi-Quan Luo. Entropic distribution matching for supervised fine-tuning of llms: Less overfitting and better diversity. In *NeurIPS 2024 Workshop on Fine-Tuning in Modern Machine Learning: Principles and Scalability*, 2024.

[20] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[21] Jan R Magnus and Heinz Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019.

[22] Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. *Advances in Neural Information Processing Systems*, 37:8901–8929, 2024.

[23] Zhixuan Pan, Shaowen Wang, and Jian Li. Understanding llm behaviors via compression: Data generation, knowledge acquisition and scaling laws. *arXiv preprint arXiv:2504.09597*, 2025.

[24] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

[25] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *Advances in neural information processing systems*, 30, 2017.

[26] Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

[27] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

[28] Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. *ICML*, 2025.

[29] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[30] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5. Ieee, 2015.

[31] 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.

[32] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[33] Haoyu Wang, Yujia Fu, Zhu Zhang, Shuo Wang, Zirui Ren, Xiaorong Wang, Zhili Li, Chaoqun He, Bo An, Zhiyuan Liu, et al. Llm $\times$ mapreduce-v2: Entropy-driven convolutional test-time scaling for generating long-form articles from extremely long resources. *arXiv preprint arXiv:2504.05732*, 2025.

[34] Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025.

[35] Cong Xu, Zhangchi Zhu, Mo Yu, Jun Wang, Jianyong Wang, and Wei Zhang. Are llm-based recommenders already the best? simple scaled cross-entropy unleashes the potential of traditional sequential recommenders. *arXiv preprint arXiv:2408.14238*, 2024.

[36] Mingjia Yin, Chuhan Wu, Yufei Wang, Hao Wang, Wei Guo, Yasheng Wang, Yong Liu, Ruiming Tang, Defu Lian, and Enhong Chen. Entropy law: The story behind data compression and llm performance. *arXiv preprint arXiv:2407.06645*, 2024.

A Proof of Theorem 1

Proof. Consider the first-order Taylor approximations:

$$u = X_{l+1}^{t+1} \approx u_0 + J_{l+1}\,\delta x, \tag{26}$$

$$v = \left(X_l^{t+1}, X_{l+1}^t\right) \approx v_0 + \begin{pmatrix} J_l \\ J_{l+1} \end{pmatrix} \delta x, \tag{27}$$

where $\delta x \sim \mathcal{N}(0, \sigma^2 I)$ is a small Gaussian perturbation around the point $x = (X_l^t, X_{l+1}^t)$. Since both u and v are affine transformations of a Gaussian random variable, they are jointly Gaussian.

From this, the joint covariance matrices are:

$$\Sigma_{uu} = \mathrm{Cov}(u) = \sigma^2 J_{l+1} J_{l+1}^\top, \tag{28}$$

$$\Sigma_{vv} = \mathrm{Cov}(v) = \sigma^2 \begin{pmatrix} J_l J_l^\top & J_l J_{l+1}^\top \\ J_{l+1} J_l^\top & J_{l+1} J_{l+1}^\top \end{pmatrix}, \tag{29}$$

$$\Sigma_{uv} = \mathrm{Cov}(u, v) = \sigma^2 J_{l+1} \begin{pmatrix} J_l \\ J_{l+1} \end{pmatrix}^\top. \tag{30}$$

The conditional entropy of a Gaussian random variable is given by [7]:

$$H(u \mid v) = \frac{1}{2} \ln\left((2\pi e)^{d_{l+1}} \det \Sigma_{u|v}\right), \tag{31}$$

$$\text{where } \Sigma_{u|v} = \Sigma_{uu} - \Sigma_{uv}\Sigma_{vv}^{-1}\Sigma_{vu}. \tag{32}$$

Now define the matrix:

$$A := \Sigma_{uu}^{-1/2}\Sigma_{uv}\Sigma_{vv}^{-1}\Sigma_{vu}\Sigma_{uu}^{-1/2}, \tag{33}$$

which is a symmetric positive semi-definite matrix quantifying how much the conditional variance is reduced by observing v.

Using the identity for positive definite matrices:

$$\Sigma_{u|v} = \Sigma_{uu}^{1/2}(I - A)\Sigma_{uu}^{1/2},$$

we obtain:

$$\det(\Sigma_{u|v}) = \det(\Sigma_{uu}) \cdot \det(I - A). \tag{34}$$

Hence, the conditional entropy becomes:

$$H(u \mid v) = \frac{1}{2} \ln\left((2\pi e)^{d_{l+1}} \det(\Sigma_{uu}) \cdot \det(I - A)\right) \tag{35}$$

$$= \frac{d_{l+1}}{2} \ln(2\pi e) + \frac{1}{2} \ln\det(\Sigma_{uu}) + \frac{1}{2} \ln\det(I - A). \tag{36}$$

Recall that $\Sigma_{uu} = \sigma^2 J_{l+1} J_{l+1}^\top$, so:

$$\ln\det(\Sigma_{uu}) = d_{l+1} \ln\sigma^2 + \ln\det(J_{l+1} J_{l+1}^\top), \tag{37}$$

where the second term can be absorbed into constants not affecting the dependency on noise.

Assuming A is small (i.e., v is weakly informative about u), we use the matrix log approximation [21]:

$$\ln\det(I - A) \approx -\mathrm{tr}(A), \tag{38}$$

yielding:

$$H(u \mid v) \approx \frac{d_{l+1}}{2}\ln(2\pi e\,\sigma^2) - \frac{1}{2}\mathrm{tr}(A). \tag{39}$$

Finally, by identifying $u = X_{l+1}^{t+1}$ and $v = (X_l^{t+1}, X_{l+1}^t)$, we obtain the entropy rate approximation:

$$H\big(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t\big) \approx \frac{d_{l+1}}{2}\ln\big(2\pi e\,\sigma^2\big) - \frac{1}{2}\mathrm{tr}(A). \tag{40}$$

□

B Proof of Theorem 2

Proof. From Theorem 1 and the Gaussian conditional entropy, we have

$$H\big(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t\big) \approx \frac{d_{l+1}}{2}\ln\big(2\pi e\,\sigma^2\big) - \frac{1}{2}\,\mathrm{tr}(A), \tag{41}$$

where

$$A = (\Sigma_{uu})^{-1/2}\,\Sigma_{uv}\,\Sigma_{vv}^{-1}\,\Sigma_{vu}\,(\Sigma_{uu})^{-1/2}. \tag{42}$$

In practice we replace the unknown δx by finite differences between consecutive batch samples. For $i = 1, \ldots, B-1$,

$$\begin{aligned} J_l(x)\,\delta x &\approx \mathbf{z}_{l,i+1} - \mathbf{z}_{l,i} = \Delta\mathbf{z}_{l,i}, \\ J_{l+1}(x)\,\delta x &\approx \mathbf{z}_{l+1,i+1} - \mathbf{z}_{l+1,i} = \Delta\mathbf{z}_{l+1,i}. \end{aligned} \tag{43}$$

where $\mathbf{z}_{l,i}$ and $\mathbf{z}_{l+1,i}$ denote the embeddings at layer l and $l+1$ on the i-th sample of a batch of size B. We treat batch index i as approximating discrete time steps, i.e., $i \approx t$.

Now recall from above that:

$$\Sigma_{uu} = \sigma^2 J_{l+1} J_{l+1}^\top, \tag{44}$$

$$\Sigma_{vv} = \sigma^2 \begin{pmatrix} J_l J_l^\top & J_l J_{l+1}^\top \\ J_{l+1} J_l^\top & J_{l+1} J_{l+1}^\top \end{pmatrix}, \tag{45}$$

$$\Sigma_{uv} = \sigma^2 J_{l+1} \begin{pmatrix} J_l \\ J_{l+1} \end{pmatrix}^\top. \tag{46}$$

When computing the conditional covariance and its inverse or whitening operations, the common factor σ^2 appears uniformly in all covariance matrices. Consequently, it cancels out naturally from all subsequent normalized or whitening expressions, so we just ignore σ^2 from now on.

Under the rank-one approximation (plus a small ridge ϵI for stability), the local covariance blocks satisfy

$$\Sigma_{uu} \approx \Delta\mathbf{z}_{l+1,i}\,\Delta\mathbf{z}_{l+1,i}^\top + \epsilon I, \tag{47}$$

$$\Sigma_{uv} \approx \Delta\mathbf{z}_{l+1,i}\,\Delta\mathbf{z}_{l,i}^\top, \tag{48}$$

$$\Sigma_{vv} \approx \begin{pmatrix} \Delta\mathbf{z}_{l,i}\,\Delta\mathbf{z}_{l,i}^\top & \Delta\mathbf{z}_{l,i}\,\Delta\mathbf{z}_{l+1,i}^\top \\ \Delta\mathbf{z}_{l+1,i}\,\Delta\mathbf{z}_{l,i}^\top & \Delta\mathbf{z}_{l+1,i}\,\Delta\mathbf{z}_{l+1,i}^\top \end{pmatrix} + \epsilon I. \tag{49}$$

Whitening involves multiplying by the inverse square root of Σ_{uu}:

$$\Sigma_{uu}^{-1/2} \approx (\Delta\mathbf{z}_{l+1,i}\Delta\mathbf{z}_{l+1,i}^\top + \epsilon I)^{-1/2}. \tag{50}$$

Since $\Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l+1,i}^\top$ is rank-one, we can use the identity for a small ridge $\epsilon > 0$:

$$\begin{aligned} & (\Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l+1,i}^\top + \epsilon I)^{-1/2} \\ \approx \quad & \frac{1}{\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon} \Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l+1,i}^\top. \end{aligned} \tag{51}$$

Similarly, the inverse covariance Σ_{vv}^{-1} is approximated by whitening the block-diagonal terms independently, yielding a block-diagonal approximation:

$$\Sigma_{vv}^{-1} \quad \approx \quad \begin{pmatrix} (\|\Delta \mathbf{z}_{l,i}\|^2 + \epsilon)^{-1} I & 0 \\ 0 & (\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon)^{-1} I \end{pmatrix}.$$

Now, substituting these approximations into the definition of A, we have explicitly:

$$\begin{aligned} A \quad = \quad & \Sigma_{uu}^{-1/2} \Sigma_{uv} \Sigma_{vv}^{-1} \Sigma_{vu} \Sigma_{uu}^{-1/2} \\ \approx \quad & \frac{\Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l+1,i}^\top}{\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon} (\Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l,i}^\top) \\ & \cdot \begin{pmatrix} (\|\Delta \mathbf{z}_{l,i}\|^2 + \epsilon)^{-1} I & 0 \\ 0 & (\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon)^{-1} I \end{pmatrix} \\ & \cdot (\Delta \mathbf{z}_{l,i} \Delta \mathbf{z}_{l+1,i}^\top) \frac{\Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l+1,i}^\top}{\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon}. \end{aligned} \tag{52}$$

Since this product is rank-one, the trace simplifies greatly. Noting the cyclic property of the trace, we have:

$$\begin{aligned} \operatorname{tr}(A) \quad \approx \quad & \frac{\Delta \mathbf{z}_{l+1,i}^\top \Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l,i}^\top \Delta \mathbf{z}_{l,i} \Delta \mathbf{z}_{l+1,i}^\top \Delta \mathbf{z}_{l+1,i} \Delta \mathbf{z}_{l+1,i}}{(\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon)^2 (\|\Delta \mathbf{z}_{l,i}\|^2 + \epsilon)} \\ = \quad & \frac{\|\Delta \mathbf{z}_{l+1,i}\|^2 \, (\Delta \mathbf{z}_{l,i}^\top \Delta \mathbf{z}_{l+1,i})^2}{(\|\Delta \mathbf{z}_{l+1,i}\|^2 + \epsilon)^2 (\|\Delta \mathbf{z}_{l,i}\|^2 + \epsilon)}. \end{aligned} \tag{53}$$

By assuming that ϵ is sufficiently small compared to embedding norms, we simplify this further to:

$$\operatorname{tr}(A) \approx \frac{\langle \Delta \mathbf{z}_{l+1,i}, \Delta \mathbf{z}_{l,i} \rangle^2}{\|\Delta \mathbf{z}_{l+1,i}\|^2 \, \|\Delta \mathbf{z}_{l,i}\|^2 + \epsilon} = \cos^2(\Delta \mathbf{z}_{l+1,i}, \Delta \mathbf{z}_{l,i}), \tag{54}$$

where we used the definition of cosine similarity:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}. \tag{55}$$

Finally, averaging over $i = 1, \ldots, B-1$ yields

$$\begin{aligned} & H\big(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t\big) \approx \frac{d_{l+1}}{2} \ln\big(2\pi e \, \sigma^2\big) \\ & - \frac{1}{2(B-1)} \sum_{i=1}^{B-1} \cos^2\big(\Delta \mathbf{z}_{l+1,i}, \, \Delta \mathbf{z}_{l,i}\big). \end{aligned} \tag{56}$$

□

C Computational and Memory Complexity

We analyze the computational complexity, memory usage, and storage costs of three methods for estimating the entropy rate of a neural network layer: (1) the exact entropy computation based on discrete probability distributions, (2) the first-order linear–Gaussian approximation (Theorem 1), and (3) the cosine-similarity-based approximation (Theorem 2).

The exact entropy rate is defined as:

$$H(X_{l+1}^{t+1} \mid X_l^{t+1}, X_{l+1}^t) = -\sum_x P(x_{l+1}^{t+1}, x_l^{t+1}, x_{l+1}^t) \log \frac{P(x_{l+1}^{t+1}, x_l^{t+1}, x_{l+1}^t)}{P(x_l^{t+1}, x_{l+1}^t)}. \tag{57}$$

To compute this expression exactly, one must estimate the full joint distribution over the discretized states of three layer representations, $x_{l+1}^{t+1}, x_l^{t+1}, x_{l+1}^t$). Assuming each activation vector is quantized into Q bins per dimension and the layer dimensionality is d, the joint distribution spans Q^{3d} states. This results in an exponential computational and memory complexity of $O(Q^{3d})$, rendering the method intractable for high-dimensional layers. Even for moderate dimensions (e.g., $d = 10$ and $Q = 10$), the state space already exceeds 10^{30}. Moreover, storing joint histograms across M batches during training incurs a total storage cost of $O(MQ^{3d})$.

The first-order linear–Gaussian approximation (Theorem 1) models the conditional entropy using local Jacobians and covariance blocks around each data point. The key operations include matrix multiplications, inversions, and square roots of $d \times d$ matrices, yielding a per-batch computational cost of $O(d^3 + d^2 B)$, where B is the batch size. Memory usage per batch is $O(d^2)$ due to storage of local covariance blocks. When averaged across M batches, the total storage requirement is $O(Md^2)$, making this method scalable to modern deep networks.

The cosine-similarity approximation (Theorem 2) avoids explicit estimation of covariance structures by using the cosine of finite differences between adjacent layer embeddings. This reduces computation to a linear cost of $O(dB)$ per batch and memory usage to $O(dB)$. Since it only requires pairwise vector operations, it scales efficiently with both batch size and layer dimension. Epoch-level storage is also efficient at $O(MB)$.

Table 2 summarizes the asymptotic complexity of the three methods. The exact computation, while theoretically accurate, is infeasible in practice. In contrast, the first-order and cosine-similarity approximations provide tractable alternatives for analyzing entropy dynamics in large-scale neural models.

Table 2: Asymptotic complexity of entropy rate estimation methods. Here, d is the layer dimension, B is the batch size, M is the number of batches, and Q is the quantization level.

Method	**Computation**	**Memory**	**Storage**
Exact	$O(Q^{3d})$	$O(Q^{3d})$	$O(MQ^{3d})$
Theorem 1	$O(d^3 + d^2 B)$	$O(d^2)$	$O(Md^2)$
Theorem 2	$O(dB)$	$O(dB)$	$O(MB)$

D Simplified Entropy-Rate Proxy

D.1 Drop the Layer-Constant Term

For

$$\overline{H}_{l+1} = \underbrace{\frac{d_{l+1}}{2} \ln\!\big(2\pi e\, \sigma^2\big)}_{C_{l+1}} - \frac{1}{2M(B-1)} \sum_{m=1}^{M} \sum_{i=1}^{B-1} \cos^2\!\big(\Delta \mathbf{z}_{l+1,i}^{(m)},\, \Delta \mathbf{z}_{l,i}^{(m)}\big), \tag{58}$$

standard Transformers have layer-invariant width $d_{l+1} \equiv d$ and we use a common noise level σ^2, so

$$C_{l+1} \equiv C = \frac{d}{2} \ln(2\pi e\, \sigma^2) \tag{59}$$

is independent of l. Define

$$S_\ell := \frac{1}{M(B-1)} \sum_{m,i} \cos^2\!\big(\Delta \mathbf{z}_{\ell,i}^{(m)},\, \Delta \mathbf{z}_{\ell-1,i}^{(m)}\big). \tag{60}$$

Then, for any a, b,

$$\overline{H}_a \le \overline{H}_b \iff C - \tfrac{1}{2} S_a \le C - \tfrac{1}{2} S_b \tag{61}$$

$$\iff S_a \ge S_b, \tag{62}$$

so ranking layers depends only on the second term. Use the centered proxy

$$\widetilde{H}_{l+1} := -\frac{1}{2M(B-1)} \sum_{m,i} \cos^2\big(\Delta \mathbf{z}_{l+1,i}^{(m)},\, \Delta \mathbf{z}_{l,i}^{(m)}\big), \tag{63}$$

which equals $\overline{H}_{l+1}$ up to the same constant C.

D.2 Positive Monotone Reparameterization

Since

$$\widetilde{H}_{l+1} = -\alpha\, S_{l+1}, \qquad \alpha := \frac{1}{2M(B-1)} > 0, \tag{64}$$

minimizing $\widetilde{H}_{l+1}$ is equivalent to maximizing S_{l+1}. Define the positive score

$$\begin{aligned} A_{l+1} &:= -\widetilde{H}_{l+1} \\ &= \alpha\, S_{l+1} \\ &= \frac{1}{2M(B-1)} \sum_{m,i} \cos^2\big(\Delta \mathbf{z}_{l+1,i}^{(m)},\, \Delta \mathbf{z}_{l,i}^{(m)}\big), \end{aligned}$$

which yields identical layer rankings and is often more interpretable (larger $A_{l+1} \Rightarrow$ less new information, more prunable).

D.3 Name and Interpretation

Because larger A_{l+1} corresponds to *smaller* conditional entropy (the layer adds less new information), we rename it for clarity as the *Redundancy Score*:

$$R_{l+1} := A_{l+1}, \tag{65}$$

so that

$$R_{l+1} = \frac{1}{2M(B-1)} \sum_{m,i} \cos^2\big(\Delta \mathbf{z}_{l+1,i}^{(m)},\, \Delta \mathbf{z}_{l,i}^{(m)}\big) \tag{66}$$

We will therefore refer to R_{l+1} as the *Redundancy Score* (higher values indicate greater redundancy and, thus, a more prunable layer).