ALGORITHMIC CAPTURE, COMPUTATIONAL COMPLEXITY, AND INDUCTIVE BIAS OF INFINITE TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We formally define algorithmic capture of combinatorial tasks as the ability of a transformer to extrapolate to arbitrary task sizes with controllable error and logarithmic sample adaptation, providing a sharp scaling criterion for distinguishing logic internalization from statistical interpolation. Empirically, across scaling ranges spanning up to 2.5 orders of magnitude, we observe evidence of capture and non-capture. By analyzing infinite-width transformers in both the lazy and rich regimes, we derive upper bounds on the inference-time computational complexity of the combinatorial tasks these networks can capture. We show that, despite their universal expressivity, transformers possess an inductive bias that disfavors higher-complexity algorithmic procedures within the efficient polynomial-time heuristic scheme class, consistent with successful capture on simpler combinatorial tasks such as induction heads, sorting, and string matching.

1 INTRODUCTION

A central yet unresolved question in the study of large language models (LLMs) is the extent to which the procedures they execute involve genuine "understanding" or merely exploit statistical correlations to interpolate a domain. Empirical work has often highlighted the fragility of this "understanding"; for instance, the GSM-Symbolic benchmark (Mirzadeh et al., 2025) shows that performance on mathematical reasoning tasks can degrade significantly when symbolic templates are altered, suggesting a reliance on pattern matching rather than robust algorithmic execution. Still, the loose and somewhat philosophical nature of these terms hinders rigorous progress. One area where statistical learning can be clearly discerned from logic internalization is that of combinatorial tasks (Sec. 3.1). Here, extrapolation to arbitrarily large task instance sizes is an inherent requirement. Such strong out-of-distribution (OOD) settings make statistical correlation and simple interpolation far less effective (Xu et al., 2020). Furthermore, existing algorithms provide a reliable ground truth regarding worst-case, average-case, and heuristic complexity. These theoretical bounds can be contrasted with the computation required for network inference, allowing us to refute the possibility that a network "grokked" (Power et al., 2022) a task without needing to interpret its inner workings. Finally, transformers – the core technology behind LLMs – can operate on variable sequence lengths, naturally lending themselves to this scalable setting once we align instance size with context length T.

In this work, we analyze whether the inductive biases of transformers inhibit or promote combinatorial task understanding. We purposely set aside the potential yet not at all obvious (Dong et al., 2021; Gromov et al., 2025) benefits of scaling depth and scratchpads to focus on the complexity of next-token predictions. To enable arbitrarily complex functions and to remove trivial computational bottlenecks associated with finite width, we consider infinitely over-parameterized networks in both the lazy and rich training regimes with an extensive scaling of width to problem size. Our key contributions are as follows.

- **Formal Definition of Combinatorial Task Capture:** We provide a verifiable definition of what it means for a neural network to capture a combinatorial task (Definition 3.1), involving an input measure that is scalable across problem sizes ($\mu_{X,T}$), an arbitrary but finite sample budget $P_0(\delta)$ to obtain correct task instance outcomes with probability $(1-\delta)$ on instances up to size T_0, and a small logarithmic budget ($O(\log(T/T_0))$) for fine-tuning on larger instance sizes ($T > T_0$).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Regime	Evaluation	Inference Complexity	Evaluation Error	EPTHS
NNGP/NTK	Kernel estimate	$O(PN_{\mathrm{MC}}T^3)$	$\sqrt{P/N_{\mathrm{MC}}}$	$O(T^{3+\epsilon})$
	Forward-pass	$O(NT^2)$	P^γ/N	$O(T^{2+\epsilon})$
Rich Mean-Field Scaling	Forward-pass	$O(NT^2)$	P^γ/N	$O(T^{2+\epsilon})$

Table 1: Summary of theoretical results. For infinite lazy transformers, the kernel predictor (NNGP/NTK) can be estimated using Monte Carlo with the specified inference complexity and error. Consequently, tasks with EPTHS complexity larger than T^3 times log factors cannot be captured ($T^{3+\epsilon}$). Approximating this predictor by a forward-pass on an extensive-width network ($N \propto P^{\gamma \geq 1}$), we obtain an improved bound on the complexity of capturable tasks under lazy learning. Making the reasonable assumption that extensive networks with rich scaling converge to their rich/mean-field limits, we obtain the same complexity bound even under strong feature learning.

- **Empirical Evidence of Captured and Non-Captured Tasks:** We identify transformers that capture the induction head, sorting, and string matching tasks. Conversely, we show several transformers, including very deep ones, that fail to capture the source-target shortest path problem, the Max Flow / Minimal Cut problem, and the context-free grammar parsing problem (Sec. 3.4).
- **Heuristic Complexity:** We introduce the efficient polynomial-time heuristic scheme (EPTHS), a distributional complexity class for combinatorial tasks that formalizes the existence of δ-correct algorithms with runtime polynomial in the instance size T (Definition 3.2). This gives rise to a task-level notion of complexity aligned with learning-based solvers and provides a principled scale against which inference-time procedures can be compared.
- **Upper Complexity Bounds:** We show that the kernel predictor of an infinite-width transformer can be estimated in $O(PN_{\mathrm{MC}}T^3)$ FLOPs with P datapoints and N_{MC} Monte Carlo (MC) samples (Theorem 4.6). Consequently, we prove that an infinite-width transformer cannot capture a combinatorial task of EPTHS complexity above $O(T^{3+\epsilon})$ (Corollary 4.7); under a finite-width convergence assumption for lazy and rich infinite-width limits (Assumption 4.9), this improves to $O(T^{2+\epsilon})$ (Corollary 4.11). Table 1 summarizes these bounds. This suggests that, while feature learning can have drastic sample complexity effects (Bietti et al., 2022; Wei et al., 2019; Ghorbani et al., 2020), it does not have similar inference complexity effects.

Our main purpose in this work is to present a framework in which computational complexity theory, OOD inductive bias of transformers, and shortcut learning can be sharply defined and contrasted. Further refinement of the upper complexity bounds is needed to explain some of our empirical results.

2 Related Work

Our work intersects with several active areas of deep learning and complexity theory.

Grokking an Algorithm. The phenomenon of "grokking" – where generalization emerges long after overfitting – has been widely observed in math tasks (Power et al., 2022; Barak et al., 2022; Nanda et al., 2023). However, these tasks have $O(\log(n)), O(1)$ computational complexity and, more crucially, experiments were carried out on specific instance sizes, rendering computational complexity ill-defined.

Kolmogorov and Sample Complexity. Prior work in algorithmic information theory suggests that neural networks are strongly biased towards functions with low Kolmogorov complexity (Valle-Pérez et al., 2019; Dingle et al., 2018). This is a different notion of complexity, which, for functions representing an algorithm, does not scale with input size. In addition, various works (Bietti et al., 2022; Ghorbani et al., 2020; Wei et al., 2019) studied the effect of feature learning on sample complexity. Lavie et al. (2024) study the sample complexity of infinite transformers. However, functions with high sample complexity are not necessarily hard to compute (e.g., k-sparse parity).

Worst-Case Expressivity / Circuit Complexity. A complementary line of work studies transformers as constant-depth threshold circuits, placing them in TC^0 (Merrill et al., 2022; Chiang et al., 2023). More recently, Chen et al. (2024) established an unconditional lower bound for multi-layer decoder-only transformers. In contrast, our perspective is "softer". We analyze average-case behavior and

learnability via infinite-width limits (NNGP/NTK), where success is approximate and often stated with high probability, and where complexity is tied to inference cost / kernel evaluation rather than to exact worst-case computation. In the limit, we have universal expressivity, and thus focus on inductive bias, whereas the aforementioned literature addresses limits to expressivity.

Length Generalization. A growing body of work studies whether transformers extrapolate to longer inputs than seen during training, particularly on algorithmic and reasoning tasks. One line of work characterizes length generalization through transformer-aligned computational models: RASP provides a formal language for transformer-computable programs (Weiss et al., 2021), and Zhou et al. (2024) conjecture length generalization when the target task admits a sufficiently simple RASP-L program. Our notion of algorithmic capture is instead task-level and black-box with respect to any particular transformer program. A second line of work studies training procedures and implicit biases, showing that standard training often fails to length-generalize on tasks such as parity or addition, while scratchpads, chain-of-thought (CoT), or curriculum learning can improve extrapolation (Abbe et al., 2024; Anil et al., 2022). Beyond fixed-horizon extrapolation, our formulation tracks the scaling of the adaptation budget with the context horizon and extends to combinatorial optimization tasks – including NP-hard problems with approximation schemes, such as the Euclidean traveling salesman problem – where failure can be related to complexity-theoretic obstructions.

Neural Algorithmic Reasoning (NAR). Neural algorithmic reasoning (NAR) has provided an empirical and conceptual foundation for studying whether neural networks can execute classical algorithms, through work on algorithmic alignment and extrapolation (Xu et al., 2020; 2021), methods based on causal regularization (Bevilacqua et al., 2023), no-hint and self-supervised training (Rodionov & Prokhorenkova, 2023), deep equilibrium formulations (Georgiev et al., 2024), and newer dynamic-programming-inspired architectures such as FloydNet (Yu et al., 2026). Complementing this line of work, we ask a task-level question: whether a transformer trained under standard supervision can maintain performance on the combinatorial problem itself as instance size grows. Our logarithmic re-adaptation criterion turns this into a black-box asymptotic test, closer to the evaluation perspective of combinatorial optimization than to imitation of a fixed canonical execution trace.

3 Problem Formulation and Main Definitions

Here we introduce our two key notions: combinatorial task capture (Definition 3.1) and efficient polynomial-time heuristic schemes (Definition 3.2). As noted in the introduction, we will show that, if an infinite-width transformer algorithmically captures a combinatorial task, then its kernel predictor places that task in an EPTHS class of complexity $O(T^{k+\epsilon})$, with $k = 3$ for any $\epsilon > 0$ (Sec. 4.3), and down to $k = 2$ under additional assumptions (Sec. 4.4). Consequently, heuristic complexity yields an obstruction to capture: if the transformer's inference-time heuristic complexity grows strictly slower than that of the combinatorial task, then the latter cannot be captured.

3.1 Combinatorial Task

In this work, we consider supervised learning of combinatorial tasks with discrete outputs. For the purposes of this paper, we use the following broad problem-oriented notion. A *combinatorial task* $g : \mathcal{X} \to \mathcal{Y}$ is a map from a space of task instances $\mathcal{X}$ to an ambient space of possible admissible outcomes $\mathcal{Y}$. To each instance $X \in \mathcal{X}$, there is an associated discrete configuration set $\mathrm{Conf}(X)$ of admissible solutions and a score function $\psi_X : \mathrm{Conf}(X) \to \mathbb{R}$. The combinatorial task is in *solution form* if $g(X) \in \mathrm{Conf}(X) \subseteq \mathcal{Y}$, where $\psi_X(g(X)) = \min_{\rho \in \mathrm{Conf}(X)} \psi_X(\rho)$, meaning $g(X)$ is a distinguished minimizer, and in *value form* if $g(X) = \min_{\rho \in \mathrm{Conf}(X)} \psi_X(\rho) \in \mathcal{Y} \subseteq \mathbb{R}$. We assume that $\mathcal{Y}$ is equipped with a task-specific metric $\mathrm{dist}_{\mathcal{Y}}$ so that $\Omega = \inf \{\mathrm{dist}_{\mathcal{Y}}(\rho, \rho') \mid \rho \neq \rho' \in \mathrm{Conf}(X) \subseteq \mathcal{Y}, X \in \mathcal{X}\} > 0$, and there is an *accuracy margin* $\Delta \geq \Omega > 0$ large enough that any prediction within $\Delta/2$ of the target implies recovery of a desired outcome. Finally, we assume a notion of task size $T = T(X) \in \mathbb{N}$ which stratifies $\mathcal{X} = \bigsqcup_{T \geq 0} \mathcal{X}_T$ into instance sets $\mathcal{X}_T$ per size T. As an example, consider the shortest path problem (see App. E.1 for details). Here, $\mathcal{X}$ is a set of finite undirected graphs with distinguished source and target nodes and X is one such. The configuration set $\mathrm{Conf}(X)$ consists of all paths from the source to the target. The task size $T(X) = V(X)$ is the number of vertices in the graph X. The task g will be in solution form if $g(X)$ is a path of minimal length, while it will be in value form when $g(X)$ is that length.

3.2 TRANSFORMER SETTING

In the transformer setting studied here, we represent an instance $X \in \mathcal{X}$ of a combinatorial task of size $T = T(X)$ by an embedded array

$$X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) \in \mathcal{X}_T \subseteq \mathbb{R}^{T \times d}, \quad \boldsymbol{x}_a \in \mathbb{R}^d, \quad a \in [T] \tag{1}$$

where d is the token embedding dimension and T plays the role of the context length and task size. Thus, throughout this paper, instances are already viewed in their embedded representation rather than as raw symbolic objects, and $\mathcal{Y}$ is chosen accordingly. We further assume that, for each size T, the instance space $\mathcal{X}_T \subseteq \mathbb{R}^{T \times d}$ is equipped with a probability measure $\mu_{X,T}$ from which training and test instances can be efficiently sampled. Where appropriate, for example, for graph combinatorial tasks, $\mu_{X,T}$ will be chosen in the critical regime, e.g., where large graph clusters emerge.

In general, X is not assumed permutable, i.e., $X_\sigma = (\boldsymbol{x}_{\sigma(1)}, \ldots, \boldsymbol{x}_{\sigma(T)})$, $\sigma \in S_T$, may represent a different task instance. In such a case we rely on positional encoding (PE). For combinatorial tasks in which all sequence indices in X are permutable, but for $O(1)$ special tokens – for example, those representing the source and target nodes in the finite undirected graph of a shortest path problem – we encode such large symmetry using PE that only distinguishes between special tokens and the rest.

Architecture-wise, we focus on decoder-only transformers with softmax self-attention (Vaswani et al., 2017). To establish notation, consider a single attention layer of a transformer block with H heads. It takes as inputs $\boldsymbol{z}'_a(X) \in \mathbb{R}^{d_{\text{model}}}$, $a \in [T]$, which we package into $Z'(X) \in \mathbb{R}^{T \times d_{\text{model}}}$. The attention layer's output at token $a \in [T]$ is then

$$\boldsymbol{z}_a(X) = \frac{1}{\sqrt{H}} \sum_{h=1}^{H} \sum_{c=1}^{T} A^h_{ac}(X) W_V^{h\top} \boldsymbol{z}'_c(X), \tag{2}$$

where

$$A^h_{ac}(X) = \text{Softmax}\left(S^h_{a\bullet}(X)\right)_c \tag{3}$$

are the attention weights and

$$S^h_{ac}(X) = \frac{1}{\sqrt{d_k}} \boldsymbol{z}'_a(X)^\top W_Q^h (W_K^h)^\top \boldsymbol{z}'_c(X) \tag{4}$$

are the attention scores for query and key weight matrices $W_Q^h, W_K^h \in \mathbb{R}^{d_{\text{model}} \times d_k}$ and value weight matrices $W_V^h \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ (see App. A.1 for further details).

3.3 TASK CAPTURE AND EPTHS

Now we are ready to rigorously define our two key notions: the algorithmic capture of a combinatorial task (Definition 3.1) and efficient polynomial-time heuristic schemes (Definition 3.2).

Definition 3.1 (Algorithmic capture of a combinatorial task)**.** A transformer f is said to *algorithmically capture* a combinatorial task $g : \mathcal{X} \to \mathcal{Y}$ if for every sufficiently small $\delta > 0$ there exist $T_0 = T_0(\delta)$, $C_0 = C_0(\delta)$, and $P_0 = P_0(\delta)$ such that, for every $T \geq T_0$,

$$\Pr_{X \sim \mu_{X,T}} \left[\text{dist}_{\mathcal{Y}}(f_T(X), g(X)) < \Delta/3\right] > 1 - \delta, \tag{5}$$

where f_T denotes the predictor obtained by the following two-stage training protocol: (1) initial training on P_0 samples from $\mu_{X,1}, \ldots, \mu_{X,T_0}$ shared across all $T \geq T_0$; and (2) a secondary fine-tuning on $C_0 \log(T/T_0)$ additional samples from $\mu_{X,1}, \ldots, \mu_{X,T}$.[1]

The choice of logarithmic data adaptation in Definition 3.1 reflects a compromise between the ideal of perfect extrapolation and the practical non-idealities of transformers. In principle, one might regard $O(1)$ additional samples as the cleanest signature of genuine capture. Empirically, however, this standard appears too strict. In combinatorial optimization problems, for instance, performance typically deteriorates as one moves away from the training size distribution; see, for instance, Kool

[1]The underlying transformer architecture, training procedure, hyperparameter choices, and initialization/training randomness convention are fixed; the probability displayed is over the test instance $X \sim \mu_{X,T}$. For a PAC-style extension of Definition 3.1, see App. A.3.

et al. (2019, Figure 5), where routing performance degrades markedly off distribution. Related work has also studied the OOD behavior of transformer variants for algorithmic computation; notably, De Luca et al. (2025) report poor OOD performance on induction-head-style tasks. We observe the same qualitative effect in our own experiments (Sec. 3.4). A natural architectural reason is attention dilution: for fixed learned key-query geometry, attention becomes increasingly diffuse as the context length grows, so even a correctly learned heuristic may require mild recalibration when transported to larger T. At the same time, allowing polynomially many new samples would blur the distinction between correcting such architectural drift and relearning the task itself. Logarithmic adaptation provides a middle ground. In particular, Edelman et al. (2022, Theorem 4.7) make a compelling theoretical case for $O(\log(T/T_0))$. Their results are consistent with our decision to formulate capture using a logarithmic re-adaptation budget rather than requiring strict zero-adaptation extrapolation.

Definition 3.2 (EPTHS)**.** An algorithm A is an *efficient polynomial-time heuristic scheme* (EPTHS) of complexity $O(T^k)$ for a combinatorial task g if for every $\delta \in (0, 1]$ there exists a δ-implementation A_δ of A that returns an output $A_\delta(X) \in \mathcal{Y}$ in time $O(\eta(1/\delta)T^k)$ such that

$$\Pr_{X\sim\mu_{X,T}}\left[\mathrm{dist}_{\mathcal{Y}}(A_\delta(X), g(X)) < \Delta/2\right] > 1 - \delta \tag{6}$$

for sufficiently large T.

In principle, the function η in Definition 3.2 is allowed to be exponential, or worse, in $1/\delta$. This is by analogy with the efficient polynomial-time approximation scheme (EPTAS), but with the guarantees reversed: whereas an EPTAS returns a $(1 + \epsilon)$-approximate solution for every input, an EPTHS provides an exact recovery of the target output for most inputs, namely with probability above $1 - \delta$ over $X \sim \mu_{X,T}$. Our use of the input measures $\mu_{X,T}$ and of heuristic complexity is inspired by the framework of distributional decision problems in average-case complexity theory, as developed by Bogdanov & Trevisan (2006). Since the tasks studied here have structured or scalar outputs rather than binary outputs, we adopt a broader task-based formulation while retaining the same distributional viewpoint.

Remark 3.3 (From average-case to EPTHS). A polynomial average-case complexity of a combinatorial task establishes an upper bound for EPTHS: if an algorithm A returns a correct outcome $g(X)$ for every $X \sim \mu_{X,T}$ in time $t(X)$ with $\mathbb{E}_{X\sim\mu_{X,T}}[t(X)] = O(T^k)$, then A establishes an EPTHS of complexity $O(T^k)$ for g, with a δ-implementation running in time $O(T^k/\delta)$. This follows from Markov's inequality (App. A.4).

3.4 Empirical Scaling Budgets on Various Tasks

Our empirical findings in Fig. 1 illustrate a spectrum of capabilities that our framework helps to decompose. We observe instances of task capture for the Induction (Panel (a)) and Sorting Vocabulary (Panel (b)) tasks. For the String Match task of a 3-token string (Panel (c)), we see capture on average, while individual seeds often, but not always, show several different slopes on our linear-log plots. In all these tasks EPTHS complexity is low and aligns well with finite circuit solutions and RASP programs (Weiss et al., 2021).

We also observe instances suggesting lack of capture in the Context-Free Grammar (CFG) task (Panel (f)), as predicted by our upper EPTHS complexity bound (Corollary 4.11). On the other hand, we find combinatorial tasks, notably the source-target shortest path problem (SPP; Panel (d)) and the Max Flow / Minimal Cut (MinCut; Panel (e)) problem, whose native EPTHS complexity is allowed by our bounds. This does not refute our theory, as an upper EPTHS complexity bound is no guarantee for capture below it. In App. E.1 we spell out each combinatorial task setting used in our experiments. Additional engineering choices are spelled out in App. E.2.

4 Theoretical Complexity Inductive Bias

Finite networks have inference-time complexity bounded by the number of FLOPs needed for their forward-pass, which, for transformers, scales as $O(LHd_{\text{model}}T^2)$ with depth L, number of attention heads H, and embedding dimension d_{model} (see Lemma A.1). Thus, any task realized by a finite-width transformer must admit inference within this $O(T^2)$ budget, up to architecture-dependent constants. Infinitely wide networks – for transformers $H, d_{\text{model}} \to \infty$ – are universal approximators (Lee et al., 2018) and, thus, expressively far richer. Still, their inductive biases, as we will show, tilt them towards

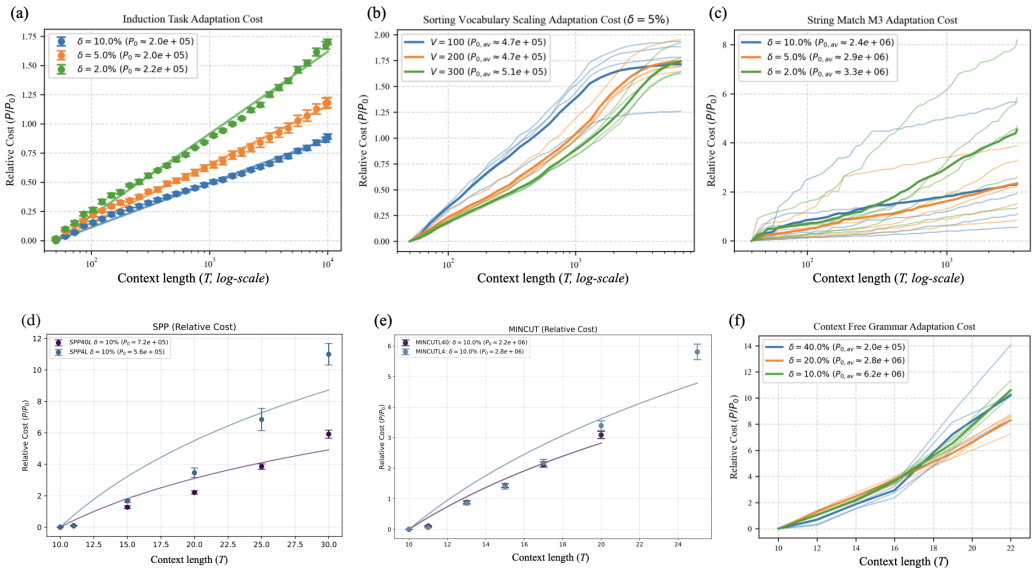


Figure 1: **Empirical Combinatorial Task Capture.** We train various models on combinatorial task instances of size T_0 (1st point on the x-axes) reaching accuracy δ after seeing $P_0(\delta)$ datapoints. We then fine-tune these models on larger instance sizes ($T > T_0$), re-achieving δ-accuracy after P extra datapoints. Panels (a)–(c) show linear-log plots of relative adaptation cost (P/P_0). Panels (a)–(b) show evidence of task capture consistent across seeds. Panel (c) shows task capture on average while individual seeds transition between different slopes. Panels (d)–(e) present results from a study of the combinatorial graph tasks, SPP and MinCut, that exhibit accelerating curves on a linear-linear scale. Solid lines are best fits to $C\log(T/T_0)$ meant to guide the eye. To mitigate expressivity concerns, depth-40 networks were used for the purple datapoints in Panels (d)–(e). For the Context-Free Grammar task in Panel (f), our theoretical bounds prohibit capture at fixed depth regardless of model capacity. Panels (a,d,e) show average empirical curves with one STD error bars together with best fits to a log. Panels (b,c,f) report individual seeds (thin lines) as well as averages (thick lines).

low inference-time computational complexity. To this end, we seek to upper bound the inference-time complexity of a trained transformer in the extreme over-parameterized limit.

4.1 Infinite-Width Predictor

We now turn to infinite transformer networks in the sense of Hron et al. (2020). In this limit, the network predictor $f(X)$ converges to a kernel predictor. At initialization, or in the context of Bayesian neural networks and SGLD training (Naveh et al., 2021; Welling & Teh, 2011), the relevant kernel is the neural network Gaussian process (NNGP) kernel (Lee et al., 2018). For gradient flow training (i.e., gradient descent at vanishing learning rate), the relevant kernel is the neural tangent kernel (NTK) (Jacot et al., 2018; Hu et al., 2020; Hron et al., 2020). In either case, the predictor takes the form of a weighted kernel expansion over the training dataset $\{X^{(\nu)}, y^{(\nu)}\}_{\nu=1}^{P}$,

$$f_K(X) = \sum_{\nu=1}^{P} \alpha_\nu K(X, X^{(\nu)}), \quad \boldsymbol{\alpha} = (K + \kappa I_P)^{-1}\boldsymbol{y}, \tag{7}$$

where $K_{\mu\nu} = K(X^{(\mu)}, X^{(\nu)})$ is the Gram matrix of dimension $P \times P$ and $\boldsymbol{y}$ is the response vector. Thus, rather than performing an infinite-compute forward-pass through the network, we may analyze the computational complexity of evaluating Eq. (7), namely, the cost of computing the P entries $K(X, X^{(\nu)})$. As we are evaluating inference-time complexity, the matrix inversion in $\boldsymbol{\alpha}$ is not incorporated; we consider it part of training complexity. Accordingly, we analyze the number of FLOPs required to compute $K(\cdot, \cdot)$.

Remark 4.1 (Kernel two-step training protocol). The two-step training protocol of Definition 3.1, which is the one used in our experiments, admits a corresponding kernel formulation (Bennani et al.,

2020). In this formulation, the first-step predictor, $f_K^{(1)}(X)$, is the kernel predictor of Eq. (7) with P_0 initial data samples. The second-step predictor, $f_K^{(2)}(X)$, is obtained by adding to $f_K^{(1)}(X)$ the kernel predictor on the second-stage P_1 labeled data samples, $\{\tilde{X}^{(\nu)}, \tilde{y}^{(\nu)}\}_{\nu=1}^{P_1}$, fitted to the residuals of the first stage,

$$f_K^{(2)}(X) = f_K^{(1)}(X) + \sum_{\nu=1}^{P_1} \tilde{\alpha}_\nu K(X, \tilde{X}^{(\nu)}),\ \tilde{\alpha}_\nu = \sum_{\mu=1}^{P_1} \left(\tilde{K} + \kappa I_{P_1}\right)^{-1}_{\nu\mu} \left(\tilde{y}^{(\mu)} - f_K^{(1)}(\tilde{X}^{(\mu)})\right), \quad (8)$$

for the second-stage Gram matrix $\tilde{K}_{\mu\nu} = K(\tilde{X}^{(\mu)}, \tilde{X}^{(\nu)})$. Here, as well, both $\boldsymbol{\alpha}$ and $\tilde{\boldsymbol{\alpha}}$ are incorporated into training-time complexity.

Remark 4.2 (Bounded kernel coefficients). We assume that the kernel coefficients $|\alpha_\nu|$ are uniformly bounded by a constant independent of P, as is typically expected. Our results in Sec. 4.3, and Proposition 4.5 in particular, also accommodate the weaker bound $|\alpha_\nu| = O(\sqrt{P})$, which corresponds to the worst case for finite-ridge regression. This is handled by scaling N_{MC}, defined in Proposition 4.5, by an additional factor of P.

4.2 Covariance Propagation

In the NNGP setting, the sufficient statistic describing the transformer block state is the token-to-token covariance matrix $\Sigma'(X_1, X_2) \in \mathbb{R}^{T\times T}$, for fixed inputs $X_1, X_2 \in \mathbb{R}^{T\times d}$, with entries

$$\Sigma'_{ab}(X_1, X_2) = \lim_{d_{\text{model}}\to\infty} \frac{1}{d_{\text{model}}} \mathbb{E}\left[\boldsymbol{z}'_a(X_1)^\top \boldsymbol{z}'_b(X_2)\right], \quad (9)$$

where $\boldsymbol{z}'_a(X)$ are the outputs of the MLP layer of the previous block (App. A.1). In the infinite-width limit, $(\boldsymbol{z}'_a(X))_{a\in[T]}$ is a centered multivariate Gaussian. Evaluating the infinite-width NNGP kernel then amounts to propagating this token-to-token covariance matrix, in the sense of Cho & Saul (2009), through the transformer blocks, starting from its attention layer. The final-block value then determines $K(X_1, X_2)$. In the infinite-width limit, we write $S(X) \in \mathbb{R}^{T\times T}$ for the limiting attention score matrix and $A(X) = \mathrm{Softmax}(S(X))$ for the corresponding limiting attention matrix.

Proposition 4.3 (Attention layer propagation, Hron et al., 2020, Theorem 3(II))**.** *Assume the incoming covariance matrix $\Sigma'(X_1, X_2)$ is given. In the infinite-width limit, the token-to-token covariance matrix propagated through the attention layer is*

$$\Sigma(X_1, X_2) = \mathbb{E}_{(S(X_1), S(X_2))}\left[A(X_1)\Sigma'(X_1, X_2)A(X_2)^\top\right]. \quad (10)$$

The proof of Proposition 4.3 uses the law of the attention score matrix $S(X)$ in the infinite-width limit as given in the following proposition.

Proposition 4.4 (Attention score distribution, Hron et al., 2020, Theorem 3(I))**.** *In the infinite-width limit, the attention score matrix $S(X)$ is a centered multivariate Gaussian, and its covariance is*

$$\mathbb{E}\left[S_{ac}(X_1)S_{be}(X_2)\right] = \Sigma'_{ab}(X_1, X_2)\Sigma'_{ce}(X_1, X_2). \quad (11)$$

Crucially, we will use the Kronecker tensor structure of the attention score covariance, made explicit in Proposition 4.4, to obtain an MC estimator of the right-hand side of Eq. (10) with efficient score sampling. We provide proofs of Propositions 4.3 and 4.4, simplified to our setting, in App. B.1 and App. A.5, respectively.

4.3 Kernel Evaluation Complexity

To analyze the computational complexity of the kernel predictor, we begin with the cost of propagating the token-to-token covariance matrix and the NTK through a transformer block, considering the cost of kernel evaluation using MC integration. Throughout this section, we fix a transformer block in the NNGP or NTK limit and assume the incoming token-to-token covariance matrix $\Sigma'(X_1, X_2) \in \mathbb{R}^{T\times T}$ of Eq. (9) or the incoming NTK

$$\Theta'_{ab}(X_1, X_2) = \lim_{d_{\text{model}}\to\infty} \frac{1}{d_{\text{model}}} \mathbb{E}\left[\nabla_\theta \boldsymbol{z}'_a(X_1)^\top \nabla_\theta \boldsymbol{z}'_b(X_2)\right], \quad a, b \in [T] \quad (12)$$

are given. We assume the coefficients α_ν of the kernel predictor induced by $\Sigma(X_1, X_2)$ and $\Theta(X_1, X_2)$ satisfy the bounded-weight assumption of Remark 4.2. In our setting, we also assume the activation function ϕ is bounded by linear growth (e.g., ReLU or GeLU).

Proposition 4.5 (Transformer block propagation)**.** *The computational complexity of evaluating the MC estimator of the predictor associated with the NNGP / NTK kernel propagated through a single transformer block is $O(PN_{\mathrm{MC}}T^3)$, with MC error scaling as $O(\sqrt{P/N_{\mathrm{MC}}})$, where N_{MC} denotes the number of MC samples.* (Proof in App. B.2.)

In App. B.3 we argue that the scaling of error and compute of Proposition 4.5 extends to the full transformer. The key observation is that, while MC error can accumulate across blocks with the noisy versions of the true input block-wise Σ' or Θ', this accumulation is T-independent since the kernel recursion relations of the attention and MLP layers admit T-independent Lipschitz constants. Thus, to maintain accuracy, one need only scale the number of MC samples with depth L, not with context length T.

Theorem 4.6 (Kernel evaluation complexity)**.** *The computational complexity of evaluating the MC estimator of the NNGP / NTK kernel predictor is $O(PN_{\mathrm{MC}}T^3)$, with MC error scaling as $O(\sqrt{P/N_{\mathrm{MC}}})$.* (Proof in App. B.3.)

Note that the computational complexity of evaluating the MC estimator will depend linearly on depth L, while the MC error will depend exponentially on L. For a finite-depth network, these are fixed constant factors, agnostic to T.

Corollary 4.7 (Kernel evaluation EPTHS)**.** *If an infinite-width transformer has algorithmically captured a combinatorial task g, then it is an EPTHS for g of complexity $O(T^{3+\epsilon})$ for any $\epsilon > 0$.*

Proof. Assume now that a transformer in the infinite-width limit has captured a combinatorial task g under $\mu_{X,T}$. Then, for sufficiently small $\delta > 0$, there exist $T_0 = T_0(\delta/2)$, $C_0 = C_0(\delta/2)$, and $P_0 = P_0(\delta/2)$ such that the kernel predictor f_δ of Eq. (8) with $P_1 = C_0 \log(T/T_0)$ adaptation data samples satisfies

$$\Pr_{X\sim\mu_{X,T}}\left[\mathrm{dist}_{\mathcal{Y}}(f_\delta(X), g(X)) < \Delta/3\right] > 1 - \delta/2, \quad \forall T \geq T_0. \tag{13}$$

Using MC integration to evaluate $f_\delta(X)$, we get $f_\delta^{\mathrm{MC}}(X)$ with the MC error scaling of Theorem 4.6 for $P = P_0 + P_1$. By the sub-Gaussian tail bound, we have

$$\Pr_{X\sim\mu_{X,T}}\left[\mathrm{dist}_{\mathcal{Y}}(f_\delta^{\mathrm{MC}}(X), f_\delta(X)) < \Delta/6\right] \geq 1 - 2e^{-c\Delta^2 N_{\mathrm{MC}}/P}. \tag{14}$$

Applying the triangle inequality, we have

$$\Pr_{X\sim\mu_{X,T}}\left[\mathrm{dist}_{\mathcal{Y}}(f_\delta^{\mathrm{MC}}(X), g(X)) < \Delta/2\right] > 1 - 2e^{-c\Delta^2 N_{\mathrm{MC}}/P} - \delta/2, \tag{15}$$

and we want $2e^{-c\Delta^2 N_{\mathrm{MC}}/P} \approx \delta/2$, which imposes $N_{\mathrm{MC}} \propto P\log(1/\delta)$. Evaluating $f_\delta^{\mathrm{MC}}(X)$, by Theorem 4.6, requires $O(PN_{\mathrm{MC}}T^3)$ computational steps. Thus, the infinite-width transformer is an EPTHS for the task it algorithmically captured with complexity $O(T^{3+\epsilon})$ w.r.t. $\mu_{X,T}$, for any $\epsilon > 0$, coming from $\log(T)^2$. Its δ-implementation is given by $f_\delta^{\mathrm{MC}}(X)$, computed in time $O(P^2\log(1/\delta)T^3)$. Notably, $\Delta/2$ provides a conclusive decision rule. □

Remark 4.8 (Polynomial data adaptation). If we were to adapt training data polynomially, say with $O(T^\kappa)$ extra samples, we could push this adaptation protocol through our MC-based kernel analysis to get an $O(T^{3+2\kappa})$ complexity bound for EPTHS for the infinite-width transformer at initialization. Our viewpoint, reflected here, is that the most reasonable separator of "understanding" from statistical interpolation is at $\kappa \to 0$.

4.4 Feed-Forward Approximation Complexity

The above results invite extensions in terms of tightness and richness. Unlike in fully connected networks (FCNs; App. D), kernel evaluation for transformers is costlier than a forward-pass, because it requires tracking correlations over the entire context. By contrast, various bounds on NTK convergence (Huang & Yau, 2020; Jacot et al., 2018; Huang et al., 2021) suggest that, at least for MLPs and CNNs, the discrepancy between the NTK kernel predictor and the finite-width network scales as P^γ/N, for some architecture-dependent constant γ, where N is the architecture width or redundancy scale: width for MLPs, channels for CNNs, or H and d_{model} for transformers.

Assumption 4.9 (NTK convergence)**.** The discrepancy between the NTK kernel predictor and the finite transformer network scales as P^γ/N, where N is the architecture redundancy scale and γ is an architecture-dependent constant.

To the best of our knowledge, how well this scaling holds in extreme OOD settings such as ours has not been fully explored yet. Finite-width approximation of the rich infinite-width NTK predictor has been studied in the literature (Bordelon & Pehlevan, 2023; Seroussi et al., 2023), potentially with $\gamma \approx 0$. We provide theoretical evidence that Assumption 4.9 nevertheless holds in the following proposition.

Proposition 4.10 (NNGP convergence bound)**.** *Consider a finite-width transformer of width N, trained with full-batch gradient descent (GD), weight decay, and additive white Gaussian noise. Let $f(X) = \boldsymbol{z}_T^{\text{out}}(X)$ for $X \in \mathbb{R}^{T\times d}$, and let $f_K(X)$ be its NNGP kernel predictor with $K = \Sigma_{TT}^{\text{out}}$. Then, in standard scaling,*

$$\left|\mathbb{E}_{f\sim P[f]}[f(X_*)] - f_K(X_*)\right| = O\left(\frac{P^\gamma}{N}\right) \tag{16}$$

*where $P[f]$ (Naveh et al., 2021, Eq. (3)) is the posterior on the trained f. (*Proof in App. C.*)*

Corollary 4.11 (Feed-forward EPTHS)**.** *Under Assumption 4.9, if an infinite-width transformer in the lazy or rich regime has algorithmically captured a combinatorial task g, then it is an EPTHS for g of complexity $O(T^{2+\epsilon})$ for any $\epsilon > 0$.*

Proof. Under Assumption 4.9, if an infinite-width transformer has captured a combinatorial task, then N need only scale poly-logarithmically with T. As a result, we can approximate the infinite-width NTK kernel predictor using a forward-pass on a finite trained network with $d_{\text{model}}, H \propto P = \log(T)^\gamma$, getting computational complexity scaling of $O(\log(T)^{2\gamma}T^2)$ by Lemma A.1. We thus get computational complexity scaling of T^2 times negligible poly-logarithmic corrections. □

5 Discussion

In this work, we moved beyond the question of what transformers can express – which, in the infinite-width limit, is effectively everything – to ask what combinatorial tasks they can capture. By formally defining algorithmic capture (Definition 3.1) as the ability to generalize to arbitrary input sizes (T) with logarithmic sample adaptation, we offered a distinction between genuine logic internalization and statistical interpolation. Our theoretical results for infinite-width transformers established an upper bound on their inference-time complexity, limiting them to the class of EPTHS (Definition 3.2) with complexity scaling no worse than $O(T^{3+\epsilon})$ for $\epsilon > 0$ (Corollary 4.7) and $O(T^{2+\epsilon})$ (Corollary 4.11) under Assumption 4.9.

The case of SPP and MinCut (Sec. 3.4) highlights a fruitful avenue of future research to refine the estimate of inference-time complexity. Our Corollary 4.7 bound relies on a "brute force" propagation of kernel covariances, while our Corollary 4.11 assumes NTK convergence (Assumption 4.9). While robust, this approach may be overly conservative. In fact, we conjecture that the dominant spectral weight of the kernel lies in the space of rather simple sparse functions.

Finally, it would be interesting to generalize these results to deeper architectures or chain of thought (CoT); the fact that these deeper nets can express complex tasks does not mean that they would learn them, as various works and our own experiments suggest. At least for lazy learning, where our bound is largely determined by the complexity of kernel evaluation, this amounts to understanding fixed points of the kernel recursion equations, which are simple for MLPs (Schoenholz et al., 2017) but remain unexplored for transformers. Turning to CoT, say in the context of reinforcement learning with verifiable rewards (RLVR) (Wen et al., 2026), our definition of capture applies verbatim, as it does not rely on the underlying inference mechanism or loss. Establishing meaningful bounds on the complexity of learnable functions using RLVR is left for future work.

AI Use Statement

In this work, we used generative AI tools for coding numerical experiments and formulating the Lipschitz proof. We have not used generative AI tools for formulating the definitions and mathematical claims, or interpreting results. Additionally, we used generative AI tools to prepare the manuscript for submission: converting it to the conference template, proofreading for typographical and grammatical errors, and formatting references. We have reviewed all AI-assisted work (e.g. proof checking, code checking, code tests, and revised texts). We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

Reproducibility Statement

Proofs of our theoretical results, together with the full transformer setup they refer to, are given in the appendix: the finite-width architecture, its inference-time complexity, and the PAC-style extension of capture in App. A.1–A.3; covariance and NTK propagation through the attention layer, the transformer block, and the full transformer (Proposition 4.5 and Theorem 4.6) in App. B; the NNGP convergence bound (Proposition 4.10) in App. C; and the fully connected case in App. D. The convergence assumption used in Sec. 4.4 is stated explicitly as Assumption 4.9. Each combinatorial task, its input distribution, and its data-generation procedure are specified in App. E.1, and model architectures, training protocols, hyperparameters, and compute are listed in App. E.2.

Ethics Statement

This work is theoretical, and its experiments use only synthetically generated combinatorial task instances. It involves no human subjects and no personal or sensitive data, and we are not aware of specific ethical concerns raised by it.

References

Emmanuel Abbe, Samy Bengio, Aryo Lotfi, and Kevin Rizk. Generalization on the unseen, logic reasoning and degree curriculum. *Journal of Machine Learning Research*, 25(331):1–58, 2024.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 38546–38556. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/fb7451e43f9c1c35b774bcfad7a5714b-Paper-Conference.pdf.

Boaz Barak, Benjamin L Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. Hidden progress in deep learning: SGD learns parities near the computational limit. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/884baf65392170763b27c914087bde01-Paper-Conference.pdf.

Mehdi Abbana Bennani, Thang Doan, and Masashi Sugiyama. Generalisation guarantees for continual learning with orthogonal gradient descent. *arXiv preprint arXiv:2006.11942*, 2020.

Dimitri P. Bertsekas. *Network Optimization: Continuous and Discrete Models*. Athena Scientific, Belmont, MA, 1998. ISBN 1-886529-02-7. URL http://www.athenasc.com/netopt.html.

Beatrice Bevilacqua, Kyriacos Nikiforou, Borja Ibarz, Ioana Bica, Michela Paganini, Charles Blundell, Jovana Mitrovic, and Petar Velickovic. Neural algorithmic reasoning with causal regularisation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pp. 2272–2288. PMLR, 2023. URL https://proceedings.mlr.press/v202/bevilacqua23a.html.

Alberto Bietti, Joan Bruna, Clayton Sanford, and Min Jae Song. Learning single-index models with shallow neural networks. *Advances in Neural Information Processing Systems*, 35: 9768–9780, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/3fb6c52aeb11e09053c16eabee74dd7b-Paper-Conference.pdf.

Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1):1–106, 2006. doi: 10.1561/0400000004.

Blake Bordelon and Cengiz Pehlevan. Dynamics of finite width kernel and prediction fluctuations in mean field neural networks, 2023. URL https://arxiv.org/abs/2304.03408.

Lijie Chen, Binghui Peng, and Hongxun Wu. Theoretical limitations of multi-layer transformer. *arXiv preprint arXiv:2412.02975*, 2024. URL https://arxiv.org/abs/2412.02975.

Zhiyi Chi. Statistical properties of probabilistic context-free grammars. *Comput. Linguist.*, 25(1): 131–160, March 1999. ISSN 0891-2017.

David Chiang, Peter Cholak, and Anand Pillay. Tighter bounds on the expressivity of transformer encoders. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202, pp. 5544–5562. PMLR, 2023. URL https://proceedings.mlr.press/v202/chiang23a.html.

Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 22, pp. 342–350, 2009. URL https://proceedings.neurips.cc/paper/2009/hash/5751ec3e9a4feab575962e78e006250d-Abstract.html.

Jesper Dall and Michael Christensen. Random geometric graphs. *Physical Review E*, 66(1):016121, 2002. doi: 10.1103/PhysRevE.66.016121.

Artur Back De Luca, George Giapitzakis, Shenghao Yang, Petar Veličković, and Kimon Fountoulakis. Positional attention: expressivity and learnability of algorithmic computation. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Kamaludin Dingle, Chico Q. Camargo, and Ard A. Louis. Input–output maps are strongly biased towards simple outputs. *Nature Communications*, 9(1):761, 2018. doi: 10.1038/s41467-018-03101-6. URL https://doi.org/10.1038/s41467-018-03101-6.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139, pp. 2793–2803. PMLR, 2021. URL https://proceedings.mlr.press/v139/dong21a.html.

Benjamin L Edelman, Surbhi Goel, Sham Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5793–5831. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/edelman22a/edelman22a.pdf.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Dip Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Dobrik Georgiev, JJ Wilson, Davide Buffelli, and Pietro Liò. Deep equilibrium algorithmic reasoning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 33638–33667. Curran Associates, Inc., 2024. doi: 10.52202/079017-1059. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3b1675de6b49cc00084374213f8c38ae-Paper-Conference.pdf.

Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. When do neural networks outperform kernel methods? *Advances in Neural Information Processing Systems*, 33: 14820–14830, 2020. URL https://proceedings.neurips.cc/paper/2020/file/a9df2255ad642b923d95503b9a7958d8-Paper.pdf.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=XJ2ZqJ2s4d. Also available as arXiv:2403.17887.

Jiri Hron, Yasaman Bahri, Jascha Sohl-Dickstein, and Roman Novak. Infinite attention: NNGP and NTK for deep attention networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pp. 4376–4386. PMLR, 2020. URL https://proceedings.mlr.press/v119/hron20a.html.

Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Hke3gyHYwH.

Jiaoyang Huang and Horng-Tzer Yau. Dynamics of deep neural networks and neural tangent hierarchy. In *International conference on machine learning*, pp. 4542–4551. PMLR, 2020.

Wei Huang, Weitao Du, and Richard Yi Da Xu. On the neural tangent kernel of deep networks with orthogonal initialization, 2021. URL https://arxiv.org/abs/2004.05867.

Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the 10th Annual Structure in Complexity Theory Conference*, pp. 134–147. IEEE, 1995. doi: 10.1109/SCT.1995.514853.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pp. 8571–8580, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/5a4be1fa34e62bb8a6ec6b91d2462f5a-Abstract.html.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9*, 2019.

Itay Lavie, Guy Gur-Ari, and Zohar Ringel. Towards understanding inductive bias in transformers: a view from infinity. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Sam Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as Gaussian processes. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/forum?id=B1EA-M-0Z.

Leonid A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986. doi: 10.1137/0215020.

William Merrill, Ashish Sabharwal, and Noah A. Smith. Saturated transformers are constant-depth threshold circuits. *Transactions of the Association for Computational Linguistics*, 10:843–856, 2022. doi: 10.1162/tacl_a_00492.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=u21fK4h5YQ. Also available as arXiv:2410.05229.

Pravin Nair. Softmax is $1/2$-Lipschitz: A tight bound across all ℓ_p norms. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=6dowaHsa6D.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/pdf?id=9XFSbDPmdW.

Gadi Naveh, Oded Ben David, Haim Sompolinsky, and Zohar Ringel. Predicting the outputs of finite deep neural networks trained with noisy gradients. *Phys. Rev. E*, 104:064301, Dec 2021. doi: 10.1103/PhysRevE.104.064301. URL https://link.aps.org/doi/10.1103/PhysRevE.104.064301.

James B. Orlin. Max flows in $O(nm)$ time, or better. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 765–774, 2013. doi: 10.1145/2488608.2488705.

Mathew Penrose. *Random Geometric Graphs*, volume 5 of *Oxford Studies in Probability*. Oxford University Press, 2003. ISBN 9780198506263.

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. URL https://arxiv.org/abs/2201.02177.

Aditya Ramamoorthy, Jun Shi, and Richard D. Wesel. On the capacity of network coding for random networks. *IEEE Transactions on Information Theory*, 51(8):2878–2885, 2005. doi: 10.1109/TIT.2005.851722.

Gleb Rodionov and Liudmila Prokhorenkova. Neural algorithmic reasoning without intermediate supervision. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 51663–51674. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a2370db7c99791ad5d9f3ef48ad6d464-Paper-Conference.pdf.

Samuel S. Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=H1W1UN9gg.

Inbar Seroussi, Gadi Naveh, and Zohar Ringel. Separation of scales and a thermodynamic description of feature learning in some CNNs. *Nature Communications*, 14(1):908, 2023. doi: 10.1038/s41467-023-36361-y. URL https://doi.org/10.1038/s41467-023-36361-y.

Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2 edition, 2006.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *ArXiv*, abs/2104.09864, 2021. URL https://api.semanticscholar.org/CorpusID:233307138.

Guillermo Valle-Pérez, Chico Q. Camargo, and Ard A. Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions, 2019. URL https://arxiv.org/abs/1805.08522.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Richard von Mises. *Mathematical Theory of Probability and Statistics*. Academic Press, New York and London, 1964.

Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. Regularization matters: Generalization and optimization of neural nets v.s. their induced kernel. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1810.05369.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers, 2021. URL https://arxiv.org/abs/2106.06981.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jGbRWwIidy.

Keyulu Xu, Jingling Li, Mozhi Zhang, Simon S. Du, Ken ichi Kawarabayashi, and Stefanie Jegelka. What can neural networks reason about? In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJxbJeHFPS.

Keyulu Xu, Mozhi Zhang, Jingling Li, Simon Shaolei Du, Ken-ichi Kawarabayashi, and Stefanie Jegelka. How neural networks extrapolate: From feedforward to graph neural networks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=UH-cmocLJC.

Jingcheng Yu, Mingliang Zeng, and Qiwei Ye. FloydNet: A learning paradigm for global relational reasoning, 2026. URL https://arxiv.org/abs/2601.19094.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua M. Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? A study in length generalization. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AssIuHnmHX.

A FINITE-WIDTH TRANSFORMERS IN DETAIL

A.1 SETUP

As stated, we focus on decoder-only transformers with softmax self-attention and no causal masking. We consider a multi-layer, multi-head transformer with L transformer blocks and H heads in each. We define each transformer block as follows. For block input $Z'(X) \in \mathbb{R}^{T \times d_{\text{model}}}$, we define the pre-activation

$$Z(X) = \frac{1}{\sqrt{H}} \sum_{h=1}^{H} A^h(X) V^h(X) \in \mathbb{R}^{T \times d_{\text{model}}} \tag{17}$$

with attention weights

$$A^h(X) = \text{Softmax}\left(\frac{1}{\sqrt{d_k}} Q^h(X) K^h(X)^\top \right) \in \mathbb{R}^{T \times T}, \tag{18}$$

where

$$Q^h(X) = Z'(X) W_Q^h \in \mathbb{R}^{T \times d_k}, \tag{19}$$

$$K^h(X) = Z'(X) W_K^h \in \mathbb{R}^{T \times d_k}, \tag{20}$$

$$V^h(X) = Z'(X) W_V^h \in \mathbb{R}^{T \times d_{\text{model}}}, \tag{21}$$

for

$$W_Q^h, W_K^h \in \mathbb{R}^{d_{\text{model}} \times d_k}, \quad W_V^h \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}, \tag{22}$$

with entries sampled

$$(W_Q^h)_{ij}, (W_K^h)_{ij}, (W_V^h)_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/d_{\text{model}}). \tag{23}$$

In particular, for $a \in [T]$,

$$\boldsymbol{z}_a(X) = \frac{1}{\sqrt{H}} \sum_{h=1}^{H} \sum_{b=1}^{T} A_{ab}^h(X) V_b^h(X) \in \mathbb{R}^{d_{\text{model}}}. \tag{24}$$

Next, we apply LayerNorm to the pre-activation per token

$$\text{LN}(\boldsymbol{z}_a) = \boldsymbol{\gamma}_{\text{LN}} \odot \frac{\boldsymbol{z}_a - \frac{1}{d_{\text{model}}} (\boldsymbol{z}_a^\top \mathbf{1}) \mathbf{1}}{\sqrt{\left\| \boldsymbol{z}_a - \frac{1}{d_{\text{model}}} (\boldsymbol{z}_a^\top \mathbf{1}) \mathbf{1} \right\|^2 / d_{\text{model}} + \epsilon}} + \boldsymbol{\beta}_{\text{LN}} \tag{25}$$

with learned parameters $\boldsymbol{\beta}_{\text{LN}}, \boldsymbol{\gamma}_{\text{LN}} \in \mathbb{R}^{d_{\text{model}}}$, where $\mathbf{1} \in \mathbb{R}^{d_{\text{model}}}$ denotes the all-ones vector, while $\odot$ denotes the Hadamard (entrywise) product. Adding multilayer perceptron (MLP) nonlinearity, the output of the transformer block at the a-th token is given by

$$\text{MLP}(\boldsymbol{z}_a) = W_{\text{MLP}} \phi(\text{LN}(\boldsymbol{z}_a)) + \boldsymbol{b}_{\text{MLP}} \tag{26}$$

for an activation function ϕ applied entry-wise and learned weights $W_{\text{MLP}} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ and $\boldsymbol{b}_{\text{MLP}} \in \mathbb{R}^{d_{\text{model}}}$ drawn via

$$W_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{\sigma_W^2}{d_{\text{model}}}\right), \qquad b_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_{\boldsymbol{b}}^2). \tag{27}$$

In this paper, we will assume the activation function ϕ is bounded by linear growth, e.g., ReLU or GeLU.

This setup is intentionally simplified: we omit the standard two-layer feed-forward network (FFN), using instead a single point-wise nonlinearity followed by a linear map. We also omit residual connections and the output projection W_O, combining heads by averaging instead, as in Eq. (17). Then

$$\theta_{\text{block}} = \left((W_Q^h, W_K^h, W_V^h)_{h=1}^H,\ \boldsymbol{\beta}_{\text{LN}},\ \boldsymbol{\gamma}_{\text{LN}},\ W_{\text{MLP}},\ \boldsymbol{b}_{\text{MLP}} \right) \tag{28}$$

denotes the collection of all learned parameters of a single transformer block in our simplified architecture.

As input to the first transformer block, $Z'(X)$ enters attention as the model-space representation of the input tokens. Including a learned embedding from raw input dimension into the model's internal representation space, we have

$$Z'(X) = XW_{\text{emb}} \tag{29}$$

for learned weights $W_{\text{emb}} \in \mathbb{R}^{d \times d_{\text{model}}}$, where we can replace X with a positionally encoded $\text{PE}(X)$, if needed.

A.2 Inference-Time Finite-Width Complexity

Lemma A.1. *The inference-time complexity of a finite-width transformer is $O(LHd_{model}T^2)$.*

Proof. First, note that the number of floating-point operations (FLOPs) involved in multiplying matrices $(m, k) \times (k, n)$ is $\approx 2mkn$. Therefore, calculating $Q(X), K(X)$ is $\approx 2Td_{\text{model}}d_k$, and calculating $V(X)$ is $\approx 2Td_{\text{model}}^2$. Thus, calculating $Q(X)K(X)^\top$ is $\approx 2T^2d_k$. Calculating Softmax is $\Theta(T^2)$, while multiplying by $V(X)$ is $\approx 2T^2d_{\text{model}}$. Thus, for the attention layer in a transformer block, we get

$$\text{FLOPs}_{\text{attn}} \approx H(4Td_{\text{model}}d_k + 2Td_{\text{model}}^2 + 2T^2d_k + \Theta(T^2) + 2T^2d_{\text{model}}). \tag{30}$$

As for the MLP block, $\text{FLOPs}_{\text{MLP}} \approx 2Td_{\text{model}}^2$. And together, we get

$$\text{FLOPs}_{\text{block}} = O(H\left[Td_{\text{model}}d_k + Td_{\text{model}}^2 + T^2(d_k + d_{\text{model}})\right]). \tag{31}$$

For a typical choice $d_k = d_{\text{model}}/H$, we get

$$\text{FLOPs}_{\text{block}} = O\left(H\left[Td_{\text{model}}^2 + T^2d_{\text{model}}\right]\right). \tag{32}$$

And for a transformer with L blocks

$$\text{FLOPs}_{\text{trans}} = O\left(LH\left[Td_{\text{model}}^2 + T^2d_{\text{model}}\right]\right), \tag{33}$$

which completes the proof. □

A.3 PAC-Style Capture

One can naturally extend Definition 3.1 to a conditional PAC-style algorithmic combinatorial task capture, in which the step-one trained model of our two-step training protocol is treated as fixed. Fix a transformer architecture, initialization distribution, training procedure, hyperparameter choices, and size-sampling schedules. For a predictor $h : \mathcal{X} \to \mathcal{Y}$, denote its size-$T$ population error by

$$R_T(h) = \Pr_{X \sim \mu_{X,T}}\left[\text{dist}_{\mathcal{Y}}(h(X), g(X)) \geq \Delta/3\right]. \tag{34}$$

We say that the transformer conditionally PAC-captures the combinatorial task $g : \mathcal{X} \to \mathcal{Y}$ if for every sufficiently small $\delta > 0$ and every $\beta > 0$ there exist $T_0 = T_0(\delta, \beta)$, $P_0 = P_0(\delta, \beta)$, and $C_0 = C_0(\delta, \beta)$ such that the following holds.

Fix an initial training set $\mathcal{D}_0$ of P_0 labeled samples from $\mu_{X,1}, \ldots, \mu_{X,T_0}$, drawn according to the fixed size-sampling schedule, and fix the randomness of the initial training stage. Let f_{T_0} be the resulting step-one trained predictor. Then, for every $T \geq T_0$, if $\mathcal{D}_T$ is a fresh adaptation set of at most $C_0 \log(T/T_0)$ labeled samples drawn from $\mu_{X,1}, \ldots, \mu_{X,T}$ according to the fixed size-sampling schedule, and f_T is obtained by fine-tuning f_{T_0} on $\mathcal{D}_T$, then

$$\Pr_{\mathcal{D}_T, \text{train}_T}\left[R_T(f_T) < \delta \,|\, \mathcal{D}_0, \text{train}_0\right] \geq 1 - \beta. \tag{35}$$

The conditional notion treats f_{T_0} as fixed and gives a PAC-style guarantee only for the logarithmic adaptation stage. A full PAC-style combinatorial task capture would then require

$$\Pr_{\mathcal{D}_0, \text{train}_0}\left[\Pr_{\mathcal{D}_T, \text{train}_T}\left[R_T(f_T) < \delta \,|\, \mathcal{D}_0, \text{train}_0\right] \geq 1 - \beta\right] \geq 1 - \alpha, \quad T \geq T_0, \tag{36}$$

with a sufficiently small $\alpha > 0$ folded in. It controls the probability that the initial training stage fails to produce a base model from which logarithmic adaptation succeeds, while β controls the failure probability of the secondary adaptation stage.

A.4 From Average-Case Complexity to EPTHS

Here we prove the claim of Remark 3.3. Let A be an algorithm that returns a correct outcome $g(X)$ for every $X \sim \mu_{X,T}$ in time $t(X)$ with $\mathbb{E}_{X\sim\mu_{X,T}}[t(X)] = O(T^k)$. Then A establishes an EPTHS of complexity $O(T^k)$ for g. By Markov's inequality

$$\Pr_{X\sim\mu_{X,T}}\left[t(X) \geq \mathbb{E}_{X\sim\mu_{X,T}}[t(X)]/\delta\right] \leq \delta. \tag{37}$$

Then, as a δ-implementation A_δ of A, run the latter on each instance X for up to $O(\mathbb{E}_{X\sim\mu_{X,T}}[t(X)]/\delta)$ steps. If A finishes within the allotted step budget, provide its outcome; otherwise, output a random outcome for the task. By construction,

$$\Pr_{X\sim\mu_{X,T}}\left[\mathrm{dist}_{\mathcal{Y}}(A_\delta(X), g(X)) = 0\right] > 1 - \delta \tag{38}$$

in time $O(T^k/\delta)$.

A.5 Attention Scores

The attention scores for the finite-width transformer were defined in Eq. (4). In the infinite-width limit, the attention score matrix is a centered multivariate Gaussian whose covariance can be calculated directly.

Proposition 4.4 (Attention score distribution, Hron et al., 2020, Theorem 3(I))**.** *In the infinite-width limit, the attention score matrix $S(X)$ is a centered multivariate Gaussian, and its covariance is*

$$\mathbb{E}\left[S_{ac}(X_1)S_{be}(X_2)\right] = \Sigma'_{ab}(X_1, X_2)\Sigma'_{ce}(X_1, X_2). \tag{11}$$

Proof. Utilizing iterated expectation,

$$\mathbb{E}\left[S^h_{ac}(X_1)S^h_{be}(X_2)\right] = \mathbb{E}_{Z'}\left[\mathbb{E}_{(W^h_Q,W^h_K)}[S^h_{ac}(X_1)S^h_{be}(X_2)|Z']\right]. \tag{39}$$

We consider the finite-width conditional overlap,

$$\begin{aligned}
&\mathbb{E}_{(W^h_Q,W^h_K)}\left[S^h_{ac}(X_1)S^h_{be}(X_2)\middle|\, Z'\right] \\
&= \frac{1}{d_k}\mathbb{E}_{(W^h_Q,W^h_K)}\left[\boldsymbol{z}'_a(X_1)^\top W^h_Q(W^h_K)^\top \boldsymbol{z}'_c(X_1)\cdot \boldsymbol{z}'_b(X_2)^\top W^h_Q(W^h_K)^\top \boldsymbol{z}'_e(X_2)\middle|\, Z'\right] \\
&= \frac{1}{d_k}\sum_{ijk}\sum_{lmn}[\boldsymbol{z}'_a]_i[\boldsymbol{z}'_c]_j[\boldsymbol{z}'_b]_l[\boldsymbol{z}'_e]_m\mathbb{E}_{(W^h_Q,W^h_K)}\left[[W^h_Q]_{ik}[W^h_K]_{jk}[W^h_Q]_{ln}[W^h_K]_{mn}\right] \\
&= \frac{1}{d_k}\sum_{ijk}\sum_{lmn}[\boldsymbol{z}'_a]_i[\boldsymbol{z}'_c]_j[\boldsymbol{z}'_b]_l[\boldsymbol{z}'_e]_m\mathbb{E}_{W^h_Q}\left[[W^h_Q]_{ik}[W^h_Q]_{ln}\right]\mathbb{E}_{W^h_K}\left[[W^h_K]_{jk}[W^h_K]_{mn}\right].
\end{aligned}$$

Independence enforces $i = l$, $j = m$, and $k = n$, thus

$$\begin{aligned}
&\mathbb{E}_{(W^h_Q,W^h_K)}\left[S^h_{ac}(X_1)S^h_{be}(X_2)\middle|\, Z'\right] \\
&= \frac{1}{d_k}\sum_{ij=1}^{d_{\text{model}}}[\boldsymbol{z}'_a]_i[\boldsymbol{z}'_b]_i[\boldsymbol{z}'_c]_j[\boldsymbol{z}'_e]_j\left(\sum_{k=1}^{d_k}\mathbb{E}_{W^h_Q}\left[[W^h_Q]^2_{ik}\right]\mathbb{E}_{W^h_K}\left[[W^h_K]^2_{jk}\right]\right) \\
&= \frac{1}{d_k}\sum_{ij=1}^{d_{\text{model}}}[\boldsymbol{z}'_a]_i[\boldsymbol{z}'_b]_i[\boldsymbol{z}'_c]_j[\boldsymbol{z}'_e]_j\left(\frac{d_k}{d^2_{\text{model}}}\right) \\
&= \left(\frac{\boldsymbol{z}'_a(X_1)^\top \boldsymbol{z}'_b(X_2)}{d_{\text{model}}}\right)\left(\frac{\boldsymbol{z}'_c(X_1)^\top \boldsymbol{z}'_e(X_2)}{d_{\text{model}}}\right),
\end{aligned}$$

where we used $(W^h_Q)_{ik}, (W^h_K)_{jk} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/d_{\text{model}})$.

In the infinite-width limit, the attention score matrix $S^h(X)$ becomes a centered multivariate Gaussian with respect to the ensemble (W^h_K, W^h_Q) as a direct consequence of the central limit theorem (CLT). The overlap concentrates to the covariance kernel

$$\lim_{d_{\text{model}}\to\infty}\mathbb{E}_{Z'}\left[\mathbb{E}_{(W^h_Q,W^h_K)}\left[S^h_{ac}(X_1)S^h_{be}(X_2)\middle|\, Z'\right]\right] = \Sigma'_{ab}(X_1, X_2)\Sigma'_{ce}(X_1, X_2), \tag{40}$$

as in Eq. (9). □

A.6 Complexity of Score Draws

The MC estimation of Eq. (10) involves sampling scores $S_{ac}(X_1)$ and $S_{be}(X_2)$ for any $a, c, b, e \in [T]$. By Proposition 4.4, the attention score matrix $S(X)$ in the infinite-width limit is a centered multivariate Gaussian with known covariance, and we can work out the computational cost of drawing a pair $(S(X_1), S(X_2))$, for fixed $X_1, X_2 \in \mathbb{R}^{T\times d}$, accordingly.

Proposition A.2. *Drawing a $2T^2$ sample $(S_{ac}(X_1), S_{be}(X_2))_{a,c,b,e\in[T]}$ admits a computational cost of $O(T^3)$.*

Proof. We utilize the Kronecker tensor structure of the score matrix covariance as well as conditioning in multivariate Gaussians to analyze its computational complexity. First, we consider $S(X_1) \in \mathbb{R}^{T\times T}$. Then

$$S(X_1) \sim \mathcal{N}(0, \Theta(X_1, X_1)) \tag{41}$$

with $\Theta(X_1, X_1) = \Sigma'(X_1, X_1) \otimes \Sigma'(X_1, X_1)$ as in Eq. (11). Note that $\Sigma'(X_1, X_1) \succeq 0$ by definition and consequently $\Theta(X_1, X_1) \succeq 0$. Given a Cholesky decomposition $\Sigma'(X_1, X_1) = L_1 L_1^\top$, we can decompose $\Theta(X_1, X_1) = (L_1 \otimes L_1)(L_1 \otimes L_1)^\top$. An $S(X_1)$ sample can then be generated at a cost of $O(T^3)$ via $S(X_1) = L_1 U L_1^\top$ for $U_{ab} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$.

Next, we apply Bayes' rule

$$p_{(S(X_1),S(X_2))} = p_{S(X_1)} p_{S(X_2)|S(X_1)}. \tag{42}$$

The joint covariance of $(S(X_1), S(X_2))$ is the block matrix

$$\begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix} = \begin{pmatrix} \Sigma'_{11} \otimes \Sigma'_{11} & \Sigma'_{12} \otimes \Sigma'_{12} \\ \Sigma'_{21} \otimes \Sigma'_{21} & \Sigma'_{22} \otimes \Sigma'_{22} \end{pmatrix} \succeq 0 \tag{43}$$

where $\Sigma'_{ij} \equiv \Sigma'(X_i, X_j)$. The conditional distribution of $S(X_2)$ given $S(X_1)$ is known (von Mises, 1964, Sec. 9.3) to be

$$S(X_2)|S(X_1) \sim \mathcal{N}(M_{\text{cond}}, \Theta_{\text{cond}}) \tag{44}$$

$$M_{\text{cond}} = \Theta_{21}\Theta_{11}^{-1}\text{vec}(S(X_1)) \tag{45}$$

$$\Theta_{\text{cond}} = \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \succeq 0 \tag{46}$$

which involves calculating

$$M_{\text{cond}} = \Sigma'_{21}(\Sigma'_{11})^{-1} S(X_1) (\Sigma'_{21}(\Sigma'_{11})^{-1})^\top \tag{47}$$

and $\Sigma'_{21}(\Sigma'_{11})^{-1}\Sigma'_{21}$ for Θ_{cond}, again, at a cost of $O(T^3)$.[2]

To draw a sample from the conditioned Gaussian, we will assume $\Sigma'_{11}, \Sigma'_{22} \succ 0$. We can then calculate

$$H := L_2^{-1}\Sigma'_{21}(\Sigma'_{11})^{-1}\Sigma'_{21}(L_2^\top)^{-1} \tag{48}$$

for a Cholesky decomposition $\Sigma'_{22} = L_2 L_2^\top$ at a cost of $O(T^3)$. Then we have

$$\left((L_2^\top)^{-1} \otimes (L_2)^{-1}\right) \Theta_{\text{cond}} \left((L_2)^{-1} \otimes (L_2^\top)^{-1}\right) = I \otimes I - H \otimes H.$$

Since the LHS is positive semi-definite (PSD), we have $I - H \succeq 0$. Consequently, if we diagonalize $H = Q\Lambda Q^{-1}$ with $\Lambda = \text{diag}(\boldsymbol{\lambda})$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_T)$, then we can generate a sample at a cost of $O(T^3)$ with

$$S(X_2)|S(X_1) = M_{\text{cond}} + L_2 Q \Delta Q^\top L_2^\top \tag{49}$$

$$\Delta := \sqrt{\mathbf{1}\mathbf{1}^\top - \boldsymbol{\lambda}\boldsymbol{\lambda}^\top} \odot Q^{-1} U (Q^\top)^{-1} \tag{50}$$

for $U_{ab} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$, where $\odot$ is the matrix element-wise product. To conclude, we obtain a sample of $(S(X_1), S(X_2))$ at an overall cost of $O(T^3)$. □

[2] If Σ'_{11} is not positive definite (PD), we can use the Moore–Penrose pseudo-inverse and the conditional becomes degenerate.

B KERNEL EVALUATION

B.1 PROPAGATING THROUGH THE ATTENTION LAYER

Proposition 4.3 (Attention layer propagation, Hron et al., 2020, Theorem 3(II))**.** *Assume the incoming covariance matrix $\Sigma'(X_1, X_2)$ is given. In the infinite-width limit, the token-to-token covariance matrix propagated through the attention layer is*

$$\Sigma(X_1, X_2) = \mathbb{E}_{(S(X_1),S(X_2))}\left[A(X_1)\Sigma'(X_1, X_2)A(X_2)^\top\right]. \tag{10}$$

Proof. In finite width, the covariance update across the attention layer requires computing the expectation $\mathbb{E}[\boldsymbol{z}_a(X_1)^\top \boldsymbol{z}_b(X_2)]$ for all pairs $a, b \in [T]$, which involves averaging over the random hidden representations $Z' = (Z'(X_1), Z'(X_2))$ and the random weights $W = (W_Q^h, W_K^h, W_V^h)$. By iterated expectation, we have

$$\mathbb{E}\left[\boldsymbol{z}_a(X_1)^\top \boldsymbol{z}_b(X_2)\right] = \mathbb{E}_{Z'}\left[\mathbb{E}_W[\boldsymbol{z}_a(X_1)^\top \boldsymbol{z}_b(X_2)|Z']\right]. \tag{51}$$

The inner expectation is straightforward

$$\begin{aligned}
&\mathbb{E}_W[\boldsymbol{z}_a(X_1)^\top \boldsymbol{z}_b(X_2)|Z'] \\
&= \frac{1}{H}\sum_{h,h'=1}^{H}\sum_{c,e=1}^{T}\mathbb{E}_W[A^h_{ac}(X_1)A^{h'}_{be}(X_2)\boldsymbol{z}'_c(X_1)^\top W_V^h W_V^{h'\top}\boldsymbol{z}'_e(X_2)|Z'] \\
&= \frac{1}{H}\sum_{h=1}^{H}\sum_{c,e=1}^{T}\boldsymbol{z}'_c(X_1)^\top \boldsymbol{z}'_e(X_2)\mathbb{E}_{(W_Q^h,W_K^h)}[A^h_{ac}(X_1)A^h_{be}(X_2)|Z'],
\end{aligned} \tag{52}$$

where, in the second equality, we used $(W_V^h)_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/d_{\text{model}})$, so that $\mathbb{E}[W_V^h W_V^{h'\top}] = \delta_{hh'} I_{d_{\text{model}}}$. Therefore,

$$\begin{aligned}
&\frac{1}{d_{\text{model}}}\mathbb{E}\left[\boldsymbol{z}_a(X_1)^\top \boldsymbol{z}_b(X_2)\right] = \\
&\qquad \sum_{c,e=1}^{T}\mathbb{E}_{Z'}\left[\frac{\boldsymbol{z}'_c(X_1)^\top \boldsymbol{z}'_e(X_2)}{d_{\text{model}}}\,\mathbb{E}_{(W_Q^h,W_K^h)}\left[\frac{1}{H}\sum_{h=1}^{H}A^h_{ac}(X_1)A^h_{be}(X_2)\,\middle|\,Z'\right]\right].
\end{aligned} \tag{53}$$

We assume the incoming token-to-token covariance matrix $\Sigma'(X_1, X_2)$ is given. Taking the limit $d_{\text{model}}, H \to \infty$, the normalized overlaps in Z' may be replaced by the deterministic $\Sigma'_{ce}(X_1, X_2)$, and the head average converges to the expectation under the limiting score law determined by Proposition 4.4. Thus, we have

$$\begin{aligned}
\Sigma_{ab}(X_1, X_2) &= \sum_{c,e=1}^{T}\Sigma'_{ce}(X_1, X_2)\mathbb{E}_{(S(X_1),S(X_2))}\left[A_{ac}(X_1)A_{be}(X_2)\right] \\
&= \mathbb{E}_{(S(X_1),S(X_2))}\left[A_{a\bullet}(X_1)\Sigma'(X_1, X_2)A_{\bullet b}(X_2)^\top\right]
\end{aligned} \tag{54}$$

which completes the proof. □

Remark B.1 (First transformer block). In the first transformer block $Z'(X)$ is the embedded input field in the model's internal representation space, which may include positional encoding (see App. A.1). This case can be analyzed similarly, but is slightly more cumbersome, since it requires working directly with the induced query, key, and value fields $Q^h(X)$, $K^h(X)$, and $V^h(X)$, rather than only with the recursive hidden field notation used for subsequent blocks.

B.2 PROPAGATING THROUGH THE TRANSFORMER BLOCK

We fix a transformer block in the NNGP or NTK limit and assume the incoming token-to-token covariance matrix $\Sigma'(X_1, X_2) \in \mathbb{R}^{T\times T}$ of Eq. (9) or the incoming NTK of Eq. (12) is given. We assume that the coefficients α_ν of the kernel predictor induced by the propagated covariance

$\Sigma(X_1, X_2)$ and NTK $\Theta(X_1, X_2)$ satisfy the bounded-weight assumption of Remark 4.2. In our setting, we also assume the activation function ϕ is bounded by linear growth (e.g., ReLU or GeLU).

Because standard transformer architectures employ LayerNorm (Eq. (25)), the input covariance matrix is bounded

$$\|\Sigma'(X_1, X_2)\|_{\max} = \max_{a,b} |\Sigma'_{ab}(X_1, X_2)| = O(1). \tag{55}$$

Unlike the covariance bound, the NTK bound does not follow directly from LayerNorm, since Θ' depends on derivatives with respect to all previous parameters. However, for fixed depth,

$$\|\Theta'(X_1, X_2)\|_{\max} = \max_{a,b} |\Theta'_{ab}(X_1, X_2)| = O(1) \tag{56}$$

follows by induction from the NTK recursion relation of Hron et al. (2020, Theorem 18),

$$\begin{aligned}\Theta_{ab} &= 2\Sigma_{ab} + \mathbb{E}_{(S_1,S_2)}\left[A_1 \Theta' A_2^\top\right]_{ab} \\ &\quad + (2\Sigma'_{ab} + \Theta'_{ab})\, \mathbb{E}_{(S_1,S_2)}\left[\mathrm{Tr}((J_S A_1^a)^\top \Sigma' J_S A_2^b (\Sigma')^\top)\right] \\ &\quad + \Sigma'_{ab} \mathbb{E}_{(S_1,S_2)}\left[\mathrm{Tr}((J_S A_1^a)^\top \Sigma' J_S A_2^b (\Theta')^\top)\right], \end{aligned} \tag{57}$$

where we used

$$\begin{aligned} A_j &\equiv A(X_j) \in \mathbb{R}^{T\times T}, \quad A_j^a = (A_j)_{a\bullet} \in \mathbb{R}^T \\ S_j &\equiv S(X_j) \in \mathbb{R}^{T\times T}, \quad j = 1, 2 \end{aligned} \tag{58}$$

and $J_S A_j^a \in \mathbb{R}^{T\times T}$ denotes the Jacobian with respect to the scores S. We omit the pair (X_1, X_2) from input and output kernels and attention scores for better readability.

Proposition 4.5 (Transformer block propagation)**.** *The computational complexity of evaluating the MC estimator of the predictor associated with the NNGP / NTK kernel propagated through a single transformer block is $O(PN_{\mathrm{MC}}T^3)$, with MC error scaling as $O(\sqrt{P/N_{\mathrm{MC}}})$, where N_{MC} denotes the number of MC samples.* (Proof in App. B.2.)

Proof. It is sufficient to show that the computational complexity of evaluating the MC estimator of the NNGP / NTK kernel propagated through the attention layer is $O(N_{\mathrm{MC}}T^3)$, with MC error scaling as $O(N_{\mathrm{MC}}^{-1/2})$. We begin with the MC estimator of the propagated token-to-token covariance. Using Proposition 4.3,

$$\Sigma(X_1, X_2) \approx \frac{1}{N_{\mathrm{MC}}} \sum_{n=1}^{N_{\mathrm{MC}}} A(X_1)^{(n)} \Sigma'(X_1, X_2) A(X_2)^{(n)\top}. \tag{59}$$

For each sample, we first draw $2T^2$ score samples $(S_{ac}(X_1), S_{be}(X_2))_{a,c,b,e\in[T]}$. By Proposition A.2, this can be done in $O(T^3)$ computational steps. Producing the attention matrices $A(X_1), A(X_2) \in \mathbb{R}^{T\times T}$ requires additional $O(T^2)$ steps. Multiplying $A(X_1)^{(n)}\Sigma'(X_1, X_2)A(X_2)^{(n)\top}$ requires $O(T^3)$ additional operations. The cost of computing a summand for a single MC sample is thus $O(T^3)$, dominated by efficient Cholesky decomposition and matrix multiplication, hence an overall $O(N_{\mathrm{MC}}T^3)$.

Next, we consider the MC estimator of the propagated NTK. By the recursion relation of Eq. (57), the computational complexity of estimating $\Theta \in \mathbb{R}^{T\times T}$ using MC integration is $O(N_{\mathrm{MC}}T^3)$ for its first two summands, as before. To work out the trace terms, we note that

$$J_S A_j^a = \mathrm{diag}(A_j^a) - A_j^a (A_j^a)^\top, \tag{60}$$

by Nair (2026, Lemma 2), and, consequently,

$$\begin{aligned}\mathrm{Tr}((J_S A_1)^\top M J_S A_2 N^\top) &= A_1(M \odot N)A_2^\top - A_1\left((MA_2^\top) \odot (NA_2^\top)\right) \\ &\quad - \left((A_1 M) \odot (A_1 N)\right) A_2^\top + \left(A_1 M A_2^\top\right) \odot \left(A_1 N A_2^\top\right), \end{aligned} \tag{61}$$

for any $M, N \in \mathbb{R}^{T\times T}$. Applying Eq. (61) once to $M = N = \Sigma'$ and twice to $M = \Sigma', N = \Theta'$, we get a computational complexity of $O(T^3)$ for the trace terms in Eq. (57) simultaneously over all $a, b \in [T]$, and hence $O(N_{\mathrm{MC}}T^3)$ for the MC estimator with N_{MC} samples.

Kernel propagation through deep FCNs (App. D) was considered by Lee et al. (2018). In our setting, the analogous covariance propagation rule is obtained by replacing the activation map ϕ with $\phi \circ \mathrm{LN}$. As we pass the covariance through the MLP layer (Eq. (26)),

$$\Sigma^{\mathrm{out}}_{ab}(X_1, X_2) = \sigma^2_{\boldsymbol{b}} + \lim_{d_{\mathrm{model}} \to \infty} \frac{\sigma^2_W}{d_{\mathrm{model}}} \mathbb{E}_{(\boldsymbol{z}_a(X_1), \boldsymbol{z}_b(X_2)) \sim \mathcal{N}(0,k)} \left[\phi\left(\mathrm{LN}(\boldsymbol{z}_a(X_1))\right)^\top \phi\left(\mathrm{LN}(\boldsymbol{z}_b(X_2))\right) \right], \quad (62)$$

where $\sigma_W, \sigma_{\boldsymbol{b}}$ are given in Eq. (27) and $\mathcal{N}(0, k)$ is the bivariate Gaussian distribution with the 2×2 covariance matrix k defined via

$$k_{aa} = \Sigma_{aa}(X_1, X_1), \quad k_{ab} = k_{ba} = \Sigma_{ab}(X_1, X_2), \quad k_{bb} = \Sigma_{bb}(X_2, X_2). \quad (63)$$

The corresponding NTK propagation through the MLP has the FCN form of Jacot et al. (2018, Theorem 1), with the scalar activation derivative kernel replaced by the Jacobian trace of $\phi \circ \mathrm{LN}$,

$$\Theta^{\mathrm{out}}_{ab}(X_1, X_2) = \Sigma^{\mathrm{out}}_{ab}(X_1, X_2) + \Theta_{ab}(X_1, X_2) \lim_{d_{\mathrm{model}} \to \infty} \frac{\sigma^2_W}{d_{\mathrm{model}}} \mathbb{E}_{(\boldsymbol{z}_a, \boldsymbol{z}_b) \sim \mathcal{N}(0,k)} \left[\mathrm{Tr}\left(J_{\boldsymbol{z}_a}\left(\phi \circ \mathrm{LN}\right)^\top J_{\boldsymbol{z}_b}\left(\phi \circ \mathrm{LN}\right) \right) \right]. \quad (64)$$

Here $J_{\boldsymbol{z}_a}(\phi \circ \mathrm{LN}) \in \mathbb{R}^{d_{\mathrm{model}} \times d_{\mathrm{model}}}$ denotes the Jacobian with respect to the coordinates of $\boldsymbol{z}_a$, namely

$$\left[J_{\boldsymbol{z}_a}(\phi \circ \mathrm{LN})\right]_{ij} = \frac{\partial [\phi(\mathrm{LN}(\boldsymbol{z}_a))]_i}{\partial z_{aj}}. \quad (65)$$

Since the MLP acts token-wise, neither Eq. (62) nor Eq. (64) mixes sequence indices. Thus, evaluating the MLP covariance and NTK updates for all $a, b \in [T]$ costs $O(T^2)$, up to a T-independent cost for the Gaussian expectations, and is dominated by the $O(T^3)$ attention-layer computation.

Each MC sample in Eq. (59) has uniformly bounded variance. Since the attention weights are non-negative and row-stochastic, each sample entry is a convex combination of the entries of $\Sigma'(X_1, X_2)$. Under LayerNorm, the $\Sigma'(X_1, X_2)$ entries are bounded by a T-independent constant (Eq. (55)). Hence, the standard MC error for the propagated covariance estimate scales as $O(N_{\mathrm{MC}}^{-1/2})$.

To control the MC variance of the NTK estimator, we use the entrywise L_1-bound for the softmax Jacobian from Lemma B.3(iv). Consequently,

$$\begin{aligned} \left|\mathrm{Tr}\left((J_S A_1^a)^\top M J_S A_2^b N^\top\right)\right| &\le \sum_{c_1, c_2, e_1, e_2} |M_{c_1 c_2}|\,|N_{e_1 e_2}|\,|(J_S A_1^a)_{c_1 e_1}|\,\left|(J_S A_2^b)_{c_2 e_2}\right| \\ &\le 4 \left\|M\right\|_{\max} \left\|N\right\|_{\max}. \end{aligned} \quad (66)$$

As noted, $\left\|\Sigma'\right\|_{\max} = O(1) = \left\|\Theta'\right\|_{\max}$, and we get that the trace terms are uniformly bounded entrywise. Hence, here, as well, the standard MC error for the propagated NTK estimate scales as $O(N_{\mathrm{MC}}^{-1/2})$.

Since Eq. (7) is a weighted sum of P independent MC-estimated kernel entries, each with fluctuation $O(N_{\mathrm{MC}}^{-1/2})$, and since the coefficients α_ν are bounded by assumption (following Remark 4.2), the total MC fluctuation of the predictor scales as $O(\sqrt{P/N_{\mathrm{MC}}})$. □

B.3 Propagating Through the Transformer

In Proposition 4.5, we established that a single MC estimation of $\Sigma(X_1, X_2)$ or $\Theta(X_1, X_2)$ yields an estimation error bounded by $O(N_{\mathrm{MC}}^{-1/2})$ irrespective of the sequence length T. However, evaluating the kernel of a deep transformer involves a sequential chain of estimations, where the empirically evaluated covariance matrix of one block is passed into the non-linear Gaussian expectations of the next. A critical theoretical concern is whether the accumulation of MC errors across transformer blocks magnifies extensively with the sequence length T. For if it does, the number of MC samples, N_{MC}, will grow polynomially or above with T in order to maintain a target accuracy at the network output, thereby invalidating the inference-complexity bound established in Proposition 4.5. Here, we show that this fortunately does not occur; the recursion maps for both the attention and MLP

layers are Lipschitz continuous with respect to the matrix max-norm, and, crucially, their Lipschitz constants are strictly $O(1)$.

As in App. B.2, we let $\Sigma'(X_1, X_2)$ and $\Theta'(X_1, X_2)$ denote the true input covariance matrix and NTK for a fixed pair of combinatorial task instance inputs $X_1, X_2 \in \mathbb{R}^{T\times d}$ of a given transformer block, while

$$\Sigma'(X_1, X_2) \mapsto \Sigma'(X_1, X_2) + \Delta\Sigma'(X_1, X_2) \tag{67}$$

$$\Theta'(X_1, X_2) \mapsto \Theta'(X_1, X_2) + \Delta\Theta'(X_1, X_2) \tag{68}$$

will be their perturbed MC estimates, with a sup-norm estimation error of

$$\|\Delta\Sigma'(X_1, X_2)\|_{\max} = \max_{a,b} |\Delta\Sigma'_{ab}(X_1, X_2)|\,, \tag{69}$$

$$\|\Delta\Theta'(X_1, X_2)\|_{\max} = \max_{a,b} |\Delta\Theta'_{ab}(X_1, X_2)|\,. \tag{70}$$

Following Eqs. (55) and (56),

$$\|\Sigma'(X_1, X_2)\|_{\max} = O(1) = \|\Theta'(X_1, X_2)\|_{\max}\,. \tag{71}$$

Let us fix a pair of tokens $a, b \in [T]$ and inputs $X_1, X_2 \in \mathbb{R}^{T\times d}$, and denote

$$\boldsymbol{s}_1 = S^{h=1}_{a\bullet}(X_1), \quad \boldsymbol{s}_2 = S^{h=1}_{b\bullet}(X_2), \quad \boldsymbol{a}(\boldsymbol{s}_i) = \mathrm{Softmax}\,(\boldsymbol{s}_i)\,. \tag{72}$$

Both $\boldsymbol{s}_i$ are considered random vectors in $\mathbb{R}^T$. Taking the expectation over $\frac{1}{H}\sum_{h=1}^{H}$ implies that we may choose a single head to evaluate on, here, chosen arbitrarily to be $h = 1$.

The $2T$-dimensional random vector $\tilde{\boldsymbol{s}} = (\boldsymbol{s}_1, \boldsymbol{s}_2)$ follows a centered multivariate Gaussian distribution $\mathcal{N}(0, C)$ with covariance matrix $C \in \mathbb{R}^{2T\times 2T}$. A perturbation introduced in the previous block, as in Eqs. (67) and (68), also shifts C to $C + \Delta C$. As we have suppressed X_1, X_2 in our notation for $\boldsymbol{s}_i$, $i = 1, 2$, we will do so for $\Sigma' \equiv \Sigma'(X_1, X_2)$ and $\Theta' \equiv \Theta'(X_1, X_2)$ for the rest of this section.

Lemma B.2 (ΔC bound)**.** $\|\Delta C\|_{\max} \le 2\,\|\Sigma'\|_{\max}\,\|\Delta\Sigma'\|_{\max} + \|\Delta\Sigma'\|^2_{\max}$.

Proof. As established in Proposition 4.4, the covariance $C \in \mathbb{R}^{2T\times 2T}$ factorizes as

$$C_{(i,c),(j,e)} = (\Sigma'_{ij})_{ab}(\Sigma'_{ij})_{ce}, \tag{73}$$

where $\Sigma'_{ij} \equiv \Sigma'(X_i, X_j)$. Under perturbation, the corresponding score covariance is shifted

$$C_{(i,c),(j,e)} \mapsto \left((\Sigma'_{ij})_{ab} + (\Delta\Sigma'_{ij})_{ab}\right)\left((\Sigma'_{ij})_{ce} + (\Delta\Sigma'_{ij})_{ce}\right), \tag{74}$$

for $i, j \in \{1, 2\}$ and $c, e \in [T]$. The difference in the score covariance matrices is then

$$\Delta C_{(i,c),(j,e)} = (\Sigma'_{ij})_{ab}(\Delta\Sigma'_{ij})_{ce} + (\Delta\Sigma'_{ij})_{ab}(\Sigma'_{ij})_{ce} + (\Delta\Sigma'_{ij})_{ab}(\Delta\Sigma'_{ij})_{ce} \tag{75}$$

and therefore $\|\Delta C\|_{\max} \le 2\,\|\Sigma'\|_{\max}\,\|\Delta\Sigma'\|_{\max} + \|\Delta\Sigma'\|^2_{\max}$. □

We will use the constraints set upon $\boldsymbol{a}(\boldsymbol{s}) \in \mathbb{R}^T$ by Softmax to establish useful bounds on its derivatives.

Lemma B.3 (Derivative norms for Softmax)**.** *For $c \in [T]$, the following bounds hold*

(i) $\|\nabla \boldsymbol{a}(\boldsymbol{s})_c\|_1 \le 2\boldsymbol{a}(\boldsymbol{s})_c$,

(ii) $\left\|\nabla^2 \boldsymbol{a}(\boldsymbol{s})_c\right\|_{1,1} \le 6\boldsymbol{a}(\boldsymbol{s})_c$,

(iii) $\left\|\nabla^3 \boldsymbol{a}(\boldsymbol{s})_c\right\|_{1,1,1} \le 26\boldsymbol{a}(\boldsymbol{s})_c$,

(iv) $\|J_{\boldsymbol{s}}\boldsymbol{a}(\boldsymbol{s})\|_{1,1} \le 2$,

where $\boldsymbol{a}(\boldsymbol{s})$ is read as either $\boldsymbol{a}(\boldsymbol{s}_i)$, $i = 1, 2$.

Proof. (i) By Nair (2026, Lemma 2), the first derivatives are given by $\partial_e \boldsymbol{a}(\boldsymbol{s})_c = \boldsymbol{a}(\boldsymbol{s})_c(\delta_{ce} - \boldsymbol{a}(\boldsymbol{s})_e)$. Then

$$\begin{aligned}\|\nabla \boldsymbol{a}(\boldsymbol{s})_c\|_1 &= \sum_{e=1}^{T} |\boldsymbol{a}(\boldsymbol{s})_c(\delta_{ce} - \boldsymbol{a}(\boldsymbol{s})_e)| \\ &= \boldsymbol{a}(\boldsymbol{s})_c(1 - \boldsymbol{a}(\boldsymbol{s})_c) + \sum_{e \neq c} \boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_e = 2\boldsymbol{a}(\boldsymbol{s})_c(1 - \boldsymbol{a}(\boldsymbol{s})_c) \leq 2\boldsymbol{a}(\boldsymbol{s})_c.\end{aligned} \tag{76}$$

(ii) The second derivatives are given by

$$\begin{aligned}\partial^2_{eb} \boldsymbol{a}(\boldsymbol{s})_c &= \partial_b \left(\boldsymbol{a}(\boldsymbol{s})_c \delta_{ce} - \boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_e\right) \\ &= \boldsymbol{a}(\boldsymbol{s})_c \delta_{ce} \delta_{cb} - \boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_b \delta_{ce} \\ &\quad - \boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_e \delta_{cb} - \boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_e \delta_{eb} + 2\boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_e \boldsymbol{a}(\boldsymbol{s})_b.\end{aligned} \tag{77}$$

Using $\sum_{e=1}^{T} \boldsymbol{a}(\boldsymbol{s})_e = 1$ and $\boldsymbol{a}(\boldsymbol{s})_e \geq 0$, the sum evaluates to

$$\sum_{e,b=1}^{T} \left|\partial^2_{eb} \boldsymbol{a}(\boldsymbol{s})_c\right| \leq \boldsymbol{a}(\boldsymbol{s})_c + \boldsymbol{a}(\boldsymbol{s})_c \cdot 1 + \boldsymbol{a}(\boldsymbol{s})_c \cdot 1 + \boldsymbol{a}(\boldsymbol{s})_c \cdot 1 + 2\boldsymbol{a}(\boldsymbol{s})_c \cdot 1 = 6\boldsymbol{a}(\boldsymbol{s})_c, \tag{78}$$

which proves the bound for $\left\|\nabla^2 \boldsymbol{a}(\boldsymbol{s})_c\right\|_{1,1}$. (iii) For the third derivatives, write

$$L_{ce} = \delta_{ce} - \boldsymbol{a}(\boldsymbol{s})_e, \tag{79}$$

so that $\partial_e \boldsymbol{a}(\boldsymbol{s})_c = \boldsymbol{a}(\boldsymbol{s})_c L_{ce}$ and $\sum_{e=1}^{T} |L_{ce}| \leq 2$. From the second-derivative formula,

$$\partial^2_{eb} \boldsymbol{a}(\boldsymbol{s})_c = \boldsymbol{a}(\boldsymbol{s})_c \left(L_{ce} L_{cb} - \boldsymbol{a}(\boldsymbol{s})_e L_{eb}\right). \tag{80}$$

Differentiating once more gives

$$\begin{aligned}\partial^3_{ebr} \boldsymbol{a}(\boldsymbol{s})_c = \boldsymbol{a}(\boldsymbol{s})_c \Big[&L_{cr}\left(L_{ce} L_{cb} - \boldsymbol{a}(\boldsymbol{s})_e L_{eb}\right) - \boldsymbol{a}(\boldsymbol{s})_e L_{er} L_{cb} - \boldsymbol{a}(\boldsymbol{s})_b L_{br} L_{ce} \\ &- \boldsymbol{a}(\boldsymbol{s})_e L_{er} L_{eb} + \boldsymbol{a}(\boldsymbol{s})_e \boldsymbol{a}(\boldsymbol{s})_b L_{br}\Big].\end{aligned} \tag{81}$$

Therefore,

$$\sum_{e,b,r=1}^{T} \left|\partial^3_{ebr} \boldsymbol{a}(\boldsymbol{s})_c\right| \leq \boldsymbol{a}(\boldsymbol{s})_c \left[12 + 4 + 4 + 4 + 2\right] = 26\boldsymbol{a}(\boldsymbol{s})_c. \tag{82}$$

Indeed, the first contribution is bounded by

$$\sum_{e,b,r=1}^{T} |L_{cr}| \left|L_{ce} L_{cb} - \boldsymbol{a}(\boldsymbol{s})_e L_{eb}\right| \leq 2 \cdot 6, \tag{83}$$

using the second-derivative bound above. The remaining four contributions are bounded by

$$\sum_{e,b,r=1}^{T} \boldsymbol{a}(\boldsymbol{s})_e |L_{er}| |L_{cb}| \leq 4, \qquad \sum_{e,b,r=1}^{T} \boldsymbol{a}(\boldsymbol{s})_b |L_{br}| |L_{ce}| \leq 4, \tag{84}$$

$$\sum_{e,b,r=1}^{T} \boldsymbol{a}(\boldsymbol{s})_e |L_{er}| |L_{eb}| \leq 4, \qquad \sum_{e,b,r=1}^{T} \boldsymbol{a}(\boldsymbol{s})_e \boldsymbol{a}(\boldsymbol{s})_b |L_{br}| \leq 2. \tag{85}$$

This proves the bound for $\left\|\nabla^3 \boldsymbol{a}(\boldsymbol{s})_c\right\|_{1,1,1}$. (iv) For the final bound,

$$\begin{aligned}\|J_{\boldsymbol{s}} \boldsymbol{a}(\boldsymbol{s})\|_{1,1} &= \sum_{c,e \in [T]} |J_{\boldsymbol{s}} \boldsymbol{a}(\boldsymbol{s})_{ce}| \\ &= \sum_{c \in [T]} \boldsymbol{a}(\boldsymbol{s})_c(1 - \boldsymbol{a}(\boldsymbol{s})_c) + \sum_{c \neq e} \boldsymbol{a}(\boldsymbol{s})_c \boldsymbol{a}(\boldsymbol{s})_e\end{aligned}$$

$$
\begin{aligned}
&= \sum_{c\in[T]} \boldsymbol{a}(\boldsymbol{s})_c(1-\boldsymbol{a}(\boldsymbol{s})_c) + \sum_{c,e\in[T]} \boldsymbol{a}(\boldsymbol{s})_c\boldsymbol{a}(\boldsymbol{s})_e - \sum_{c\in[T]} \boldsymbol{a}(\boldsymbol{s})_c^2 \\
&= \sum_{c\in[T]} \boldsymbol{a}(\boldsymbol{s})_c - \sum_{c\in[T]} \boldsymbol{a}(\boldsymbol{s})_c^2 + \left(\sum_{c\in[T]} \boldsymbol{a}(\boldsymbol{s})_c\right)^2 - \sum_{c\in[T]} \boldsymbol{a}(\boldsymbol{s})_c^2 \\
&= (1-\|\boldsymbol{a}(\boldsymbol{s})\|_2^2) + (1-\|\boldsymbol{a}(\boldsymbol{s})\|_2^2) \\
&\leq 2, \qquad (86)
\end{aligned}
$$

using Nair (2026, Lemma 2) to establish the second equality and the fact that $\boldsymbol{a}(\boldsymbol{s})$ is a probability vector. □

B.3.1 NNGP

We shall begin with the token-to-token covariance matrix. By Proposition 4.3, the exact covariance update is given by

$$
\Sigma_{ab} = \mathbb{E}_{\tilde{\boldsymbol{s}}}\left[q_1(\tilde{\boldsymbol{s}})\right], \quad q_1(\tilde{\boldsymbol{s}}) = \sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\Sigma'_{ce}\boldsymbol{a}(\boldsymbol{s}_2)_e, \qquad (87)
$$

where $\boldsymbol{a}(\boldsymbol{s}) = \mathrm{Softmax}(\boldsymbol{s})$. We can then bound its total discrepancy,

$$
\begin{aligned}
|\Delta\Sigma_{ab}| &= \left|\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\left(\Sigma'_{ce}+\Delta\Sigma'_{ce}\right)\boldsymbol{a}(\boldsymbol{s}_2)_e\right] - \Sigma_{ab}\right| \\
&\leq \underbrace{\left|\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\Sigma'_{ce}\boldsymbol{a}(\boldsymbol{s}_2)_e\right] - \Sigma_{ab}\right|}_{e^{\Sigma}_{\text{measure}}} \\
&\quad + \underbrace{\left|\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\Delta\Sigma'_{ce}\boldsymbol{a}(\boldsymbol{s}_2)_e\right]\right|}_{e^{\Sigma}_{\text{val}}}, \qquad (88)
\end{aligned}
$$

where the first error contribution, $e^{\Sigma}_{\text{measure}}$, comes from the indirect perturbative effect via $\boldsymbol{a}(\boldsymbol{s}_1)_c$ and $\boldsymbol{a}(\boldsymbol{s}_2)_e$, while the second, e^{Σ}_{val}, comes from the direct perturbative effect on Σ'_{ce}.

Lemma B.4 (Bounding e^{Σ}_{val})**.** $e^{\Sigma}_{\text{val}} \leq \|\Delta\Sigma'\|_{\max}$.

Proof. The error induced by substituting Σ' for $\Delta\Sigma'$ can be bounded with

$$
\begin{aligned}
e^{\Sigma}_{\text{val}} &= \left|\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\Delta\Sigma'_{ce}\boldsymbol{a}(\boldsymbol{s}_2)_e\right]\right| \\
&\leq \|\Delta\Sigma'\|_{\max}\,\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\left(\sum_{c=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\right)\left(\sum_{e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_2)_e\right)\right] \\
&\leq \|\Delta\Sigma'\|_{\max}. \qquad (89)
\end{aligned}
$$

The 2nd inequality results from Softmax outputs summing to 1. The error e^{Σ}_{val}, thus, absorbs the T^2 summands and is exactly 1-Lipschitz and independent of T. □

Lemma B.5 (Bounding $e^{\Sigma}_{\text{measure}}$)**.** $e^{\Sigma}_{\text{measure}} \leq \frac{1}{2}\|\Delta C\|_{\max}\sup_{\tilde{\boldsymbol{s}}\in\mathbb{R}^{2T}}\left\|\nabla^2 q_1(\tilde{\boldsymbol{s}})\right\|_{1,1}$.

Proof. Here, we wish to bound the shift in the expectation due to the perturbed distribution of $\tilde{\boldsymbol{s}}$,

$$
\begin{aligned}
&e^{\Sigma}_{\text{measure}} = \\
&\left|\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\Sigma'_{ce}\boldsymbol{a}(\boldsymbol{s}_2)_e\right] - \mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C)}\left[\sum_{c,e=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_c\Sigma'_{ce}\boldsymbol{a}(\boldsymbol{s}_2)_e\right]\right|. \qquad (90)
\end{aligned}
$$

To bound the shift in expectation under the perturbed Gaussian measure, we employ a multivariate Gaussian interpolation technique – a standard application of Gaussian integration by parts, often associated with Price's Theorem. We construct a continuous interpolation between the unperturbed covariance C and the perturbed covariance $C + \Delta C$ parameterized by $t \in [0, 1]$:

$$C(t) := C + t\Delta C, \tag{91}$$

with $\tilde{\boldsymbol{s}}(t) \sim \mathcal{N}(0, C(t))$ interpolating accordingly. We define the expected value at t to be

$$\varphi(t) = \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \tilde{\boldsymbol{s}}(t)} \left[q_1(\tilde{\boldsymbol{s}}) \right]. \tag{92}$$

The total error introduced by the measure pathway is the absolute difference at the endpoints, $e^{\Sigma}_{\text{measure}} = |\varphi(1) - \varphi(0)|$. By the fundamental theorem of calculus, this is bounded by

$$e^{\Sigma}_{\text{measure}} = \left| \int_0^1 \varphi'(t) dt \right| \leq \int_0^1 |\varphi'(t)| \, dt. \tag{93}$$

To compute $\varphi'(t)$, we utilize Price's Theorem,

$$\varphi'(t) = \frac{1}{2} \sum_{(i,c),(j,e)} \Delta C_{(i,c),(j,e)} \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \tilde{\boldsymbol{s}}(t)} \left[\frac{\partial^2 q_1(\tilde{\boldsymbol{s}})}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{j,e}} \right]. \tag{94}$$

Substituting the derivative back into the integral,

$$\begin{aligned} e^{\Sigma}_{\text{measure}} &\leq \frac{1}{2} \int_0^1 \sum_{(i,c),(j,e)} \left| \Delta C_{(i,c),(j,e)} \right| \left| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \tilde{\boldsymbol{s}}(t)} \left[\frac{\partial^2 q_1(\tilde{\boldsymbol{s}})}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{j,e}} \right] \right| dt \\ &\leq \frac{1}{2} \left\| \Delta C \right\|_{\max} \int_0^1 \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \tilde{\boldsymbol{s}}(t)} \left[\sum_{(i,c),(j,e)} \left| \frac{\partial^2 q_1(\tilde{\boldsymbol{s}})}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{j,e}} \right| \right] dt. \end{aligned} \tag{95}$$

We can now obtain a strict, distribution-free upper bound by replacing the expression inside the expectation with its supremum. Because this supremum is independent of the interpolation parameter t, both expectation and integral trivially evaluate to 1, giving us

$$e^{\Sigma}_{\text{measure}} \leq \frac{1}{2} \left\| \Delta C \right\|_{\max} \sup_{\tilde{\boldsymbol{s}} \in \mathbb{R}^{2T}} \left\{ \sum_{(i,c),(j,e)} \left| \frac{\partial^2 q_1(\tilde{\boldsymbol{s}})}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{j,e}} \right| \right\}. \tag{96}$$

Finally, we identify

$$\left\| \nabla^2 q_1(\tilde{\boldsymbol{s}}) \right\|_{1,1} = \sum_{(i,c),(j,e)} \left| \frac{\partial^2 q_1(\tilde{\boldsymbol{s}})}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{j,e}} \right|, \tag{97}$$

which completes our proof. □

Next, we bound $\left\| \nabla^2 q_1(\tilde{\boldsymbol{s}}) \right\|_{1,1}$ in Lemma B.5 by showing that each summand of Eq. (97) scales as $O(1/T^2)$. To this end, we use the constraints set upon $\boldsymbol{a}(\boldsymbol{s}) \in \mathbb{R}^T$ by Softmax, as established in Lemma B.3.

Lemma B.6 (L_1-bounded Hessian of q_1)**.** $\sup_{\tilde{\boldsymbol{s}} \in \mathbb{R}^{2T}} \left\| \nabla^2 q_1(\tilde{\boldsymbol{s}}) \right\|_{1,1} \leq 20 \left\| \Sigma' \right\|_{\max}$.

Proof. By definition,

$$\left\| \nabla^2 q_1(\tilde{\boldsymbol{s}}) \right\|_{1,1} = \sum_{i \in \{1,2\}} \sum_{c,e=1}^{T} \left| \frac{\partial^2 q_1}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{i,e}} \right| + 2 \sum_{c,e=1}^{T} \left| \frac{\partial^2 q_1}{\partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{2,e}} \right|. \tag{98}$$

For the first summand on the right-hand side, with $i = 1$,

$$\sum_{c,e=1}^{T} \left| \frac{\partial^2 q_1}{\partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{1,e}} \right| = \sum_{c,e=1}^{T} \left| \sum_{g,h=1}^{T} \frac{\partial^2 \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial (\boldsymbol{s}_1)_c \partial (\boldsymbol{s}_1)_e} \Sigma'_{gh} \boldsymbol{a}(\boldsymbol{s}_2)_h \right|$$

$$\leq \sum_{g,h=1}^{T} \left(\sum_{c,e=1}^{T} \left| \frac{\partial^2 \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial(\boldsymbol{s}_1)_c \partial(\boldsymbol{s}_1)_e} \right| \right) \left| \Sigma'_{gh} \boldsymbol{a}(\boldsymbol{s}_2)_h \right|$$

$$= \sum_{g,h=1}^{T} \left\| \nabla^2 \boldsymbol{a}(\boldsymbol{s}_1)_g \right\|_{1,1} \left| \Sigma'_{gh} \boldsymbol{a}(\boldsymbol{s}_2)_h \right|$$

$$\leq 6 \left\| \Sigma' \right\|_{\max} \left(\sum_{g=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_g \right) \left(\sum_{h=1}^{T} \boldsymbol{a}(\boldsymbol{s}_2)_h \right)$$

$$\leq 6 \left\| \Sigma' \right\|_{\max}, \tag{99}$$

applying Lemma B.3(ii) and the row-stochasticity of the attention matrix in the last two inequalities above. The same applies, of course, to the case $i = 2$. For the second summand on the right-hand side of Eq. (98), we have

$$\sum_{c,e=1}^{T} \left| \frac{\partial^2 q_1}{\partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{2,e}} \right| = \sum_{c,e=1}^{T} \left| \sum_{g,h=1}^{T} \frac{\partial \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial \boldsymbol{s}_{1,c}} \Sigma'_{gh} \frac{\partial \boldsymbol{a}(\boldsymbol{s}_2)_h}{\partial \boldsymbol{s}_{2,e}} \right|$$

$$\leq \sum_{g,h=1}^{T} \left(\sum_{c=1}^{T} \left| \frac{\partial \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial \boldsymbol{s}_{1,c}} \right| \right) \left| \Sigma'_{gh} \right| \left(\sum_{e=1}^{T} \left| \frac{\partial \boldsymbol{a}(\boldsymbol{s}_2)_h}{\partial \boldsymbol{s}_{2,e}} \right| \right)$$

$$\leq \left\| \Sigma' \right\|_{\max} \sum_{g,h=1}^{T} \left\| \nabla \boldsymbol{a}(\boldsymbol{s}_1)_g \right\|_1 \left\| \nabla \boldsymbol{a}(\boldsymbol{s}_2)_h \right\|_1$$

$$\leq 4 \left\| \Sigma' \right\|_{\max} \left(\sum_{g=1}^{T} \boldsymbol{a}(\boldsymbol{s}_1)_g \right) \left(\sum_{h=1}^{T} \boldsymbol{a}(\boldsymbol{s}_2)_h \right)$$

$$\leq 4 \left\| \Sigma' \right\|_{\max}, \tag{100}$$

applying Lemma B.3(i) and the row-stochasticity of the attention matrix again. Substituting these inequalities into Eq. (98), we get

$$\sup_{\tilde{\boldsymbol{s}} \in \mathbb{R}^{2T}} \left\| \nabla^2 q_1(\tilde{\boldsymbol{s}}) \right\|_{1,1} \leq 20 \left\| \Sigma' \right\|_{\max}, \tag{101}$$

which completes the proof. □

B.3.2 NTK

We move on to discussing the discrepancy in the NTK kernel evaluation. By the recursion relation in Eq. (57), we need only consider the trace terms. Let us denote

$$F = \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0,C)}[q_2(\tilde{\boldsymbol{s}})], \quad q_2(\tilde{\boldsymbol{s}}) = \mathrm{Tr}\left(\left(J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1) \right)^\top \Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Sigma')^\top \right), \tag{102}$$

$$G = \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0,C)}[q_3(\tilde{\boldsymbol{s}})], \quad q_3(\tilde{\boldsymbol{s}}) = \mathrm{Tr}\left(\left(J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1) \right)^\top \Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Theta')^\top \right). \tag{103}$$

We will focus on the discrepancy in F and the discrepancy in G will follow in a similar fashion. The former is given by

$$|\Delta F| = \left| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0, C + \Delta C)} \left[\mathrm{Tr}\left(\left(J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1) \right)^\top (\Sigma' + \Delta\Sigma') J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Sigma' + \Delta\Sigma')^\top \right) \right] - F \right|$$

$$\leq e^F_{\mathrm{val}} + e^F_{\mathrm{measure}} \tag{104}$$

with value and measure errors given by

$$e^F_{\mathrm{val}} = \Big| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0, C + \Delta C)} \Big[\mathrm{Tr}\left(\left(J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1) \right)^\top (\Sigma' + \Delta\Sigma') J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Sigma' + \Delta\Sigma')^\top \right) -$$

$$\mathrm{Tr}\left(\left(J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1) \right)^\top \Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Sigma')^\top \right) \Big] \Big| \tag{105}$$

and

$$e^F_{\mathrm{measure}} = \left| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0, C + \Delta C)} \left[\mathrm{Tr}\left(\left(J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1) \right)^\top \Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Sigma')^\top \right) \right] - F \right| \tag{106}$$

Lemma B.7 (Bounding e^F_{val}). $e^F_{\text{val}} \le 8 \|\Delta\Sigma'\|_{\max} \|\Sigma'\|_{\max} + 4 \|\Delta\Sigma'\|^2_{\max}$.

Proof. We begin by identifying three terms in Eq. (105).

$$e^F_{\text{val}} \le e_1 + e_2 + e_3, \tag{107}$$

with

$$e_1 = \left| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0, C+\Delta C)} \left[\operatorname{Tr}\left((J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1))^\top \Delta\Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Sigma')^\top \right) \right] \right|, \tag{108}$$

$$e_2 = \left| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0, C+\Delta C)} \left[\operatorname{Tr}\left((J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1))^\top \Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Delta\Sigma')^\top \right) \right] \right|, \tag{109}$$

$$e_3 = \left| \mathbb{E}_{\tilde{\boldsymbol{s}} \sim \mathcal{N}(0, C+\Delta C)} \left[\operatorname{Tr}\left((J_{\boldsymbol{s}_1} \boldsymbol{a}(\boldsymbol{s}_1))^\top \Delta\Sigma' J_{\boldsymbol{s}_2} \boldsymbol{a}(\boldsymbol{s}_2) (\Delta\Sigma')^\top \right) \right] \right|. \tag{110}$$

Repeating the calculation of Eq. (66), for each term, we get

$$e_1 + e_2 + e_3 \le 8 \|\Delta\Sigma'\|_{\max} \|\Sigma'\|_{\max} + 4 \|\Delta\Sigma'\|^2_{\max}, \tag{111}$$

as stated, utilizing Lemma B.3(iv). □

Lemma B.8 (Bounding e^F_{measure}). $e^F_{\text{measure}} \le \frac{1}{2} \|\Delta C\|_{\max} \sup_{\tilde{\boldsymbol{s}}} \left\| \nabla^2_{\tilde{\boldsymbol{s}}} q_2(\tilde{\boldsymbol{s}}) \right\|_{1,1}$.

Proof. The result follows from the exact same Gaussian interpolation argument as in Lemma B.5, replacing q_1 with q_2. □

Lemma B.9 (L_1-bounded Hessian of q_2). $\sup_{\tilde{\boldsymbol{s}} \in \mathbb{R}^{2T}} \left\| \nabla^2 q_2(\tilde{\boldsymbol{s}}) \right\|_{1,1} \le 176 \|\Sigma'\|^2_{\max}$.

Proof. Expanding q_2 of Eq. (102) entrywise,

$$q_2(\tilde{\boldsymbol{s}}) = \sum_{g,h,r,m=1}^{T} \Sigma'_{gh} \Sigma'_{rm} \frac{\partial \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial \boldsymbol{s}_{1,r}} \frac{\partial \boldsymbol{a}(\boldsymbol{s}_2)_h}{\partial \boldsymbol{s}_{2,m}}. \tag{112}$$

As in Eq. (98),

$$\left\| \nabla^2 q_2(\tilde{\boldsymbol{s}}) \right\|_{1,1} = \sum_{i \in \{1,2\}} \sum_{c,e=1}^{T} \left| \frac{\partial^2 q_2}{\partial \boldsymbol{s}_{i,c} \partial \boldsymbol{s}_{i,e}} \right| + 2 \sum_{c,e=1}^{T} \left| \frac{\partial^2 q_2}{\partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{2,e}} \right|. \tag{113}$$

For the first summand, with $i = 1$,

$$\begin{aligned}
\sum_{c,e=1}^{T} \left| \frac{\partial^2 q_2}{\partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{1,e}} \right| &\le \sum_{g,h,r,m=1}^{T} |\Sigma'_{gh} \Sigma'_{rm}| \left(\sum_{c,e=1}^{T} \left| \frac{\partial^3 \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial \boldsymbol{s}_{1,r} \partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{1,e}} \right| \right) \left| \frac{\partial \boldsymbol{a}(\boldsymbol{s}_2)_h}{\partial \boldsymbol{s}_{2,m}} \right| \\
&\le \|\Sigma'\|^2_{\max} \left(\sum_{g=1}^{T} \left\| \nabla^3 \boldsymbol{a}(\boldsymbol{s}_1)_g \right\|_{1,1,1} \right) \left(\sum_{h=1}^{T} \|\nabla \boldsymbol{a}(\boldsymbol{s}_2)_h\|_1 \right) \\
&\le 52 \|\Sigma'\|^2_{\max},
\end{aligned} \tag{114}$$

using Lemma B.3(i) and (iii). The same bound holds for $i = 2$. For the mixed summand,

$$\begin{aligned}
\sum_{c,e=1}^{T} \left| \frac{\partial^2 q_2}{\partial \boldsymbol{s}_{1,c} \partial \boldsymbol{s}_{2,e}} \right| &\le \sum_{g,h,r,m=1}^{T} |\Sigma'_{gh} \Sigma'_{rm}| \left(\sum_{c=1}^{T} \left| \frac{\partial^2 \boldsymbol{a}(\boldsymbol{s}_1)_g}{\partial \boldsymbol{s}_{1,r} \partial \boldsymbol{s}_{1,c}} \right| \right) \left(\sum_{e=1}^{T} \left| \frac{\partial^2 \boldsymbol{a}(\boldsymbol{s}_2)_h}{\partial \boldsymbol{s}_{2,m} \partial \boldsymbol{s}_{2,e}} \right| \right) \\
&\le \|\Sigma'\|^2_{\max} \left(\sum_{g=1}^{T} \left\| \nabla^2 \boldsymbol{a}(\boldsymbol{s}_1)_g \right\|_{1,1} \right) \left(\sum_{h=1}^{T} \left\| \nabla^2 \boldsymbol{a}(\boldsymbol{s}_2)_h \right\|_{1,1} \right) \\
&\le 36 \|\Sigma'\|^2_{\max},
\end{aligned} \tag{115}$$

using Lemma B.3(ii). Substituting these inequalities into the Hessian expansion gives

$$\left\| \nabla^2 q_2(\tilde{\boldsymbol{s}}) \right\|_{1,1} \le 52 \|\Sigma'\|^2_{\max} + 52 \|\Sigma'\|^2_{\max} + 2 \cdot 36 \|\Sigma'\|^2_{\max} = 176 \|\Sigma'\|^2_{\max}. \tag{116}$$

Taking the supremum over $\tilde{\boldsymbol{s}} \in \mathbb{R}^{2T}$ completes the proof. □

B.3.3 THEOREM 4.6

Theorem 4.6 (Kernel evaluation complexity)**.** *The computational complexity of evaluating the MC estimator of the NNGP / NTK kernel predictor is $O(PN_{\mathrm{MC}}T^3)$, with MC error scaling as $O(\sqrt{P/N_{\mathrm{MC}}})$.* (Proof in App. B.3.)

Proof. Applying Lemmas B.4, B.5, and B.6 to Eq. (88), we get

$$|\Delta\Sigma_{ab}| \leq \|\Delta\Sigma'\|_{\max} + 10\,\|\Delta C\|_{\max}\,\|\Sigma'\|_{\max}\,, \tag{117}$$

and, when we apply Lemma B.2, we get

$$|\Delta\Sigma_{ab}| \leq \|\Delta\Sigma'\|_{\max} + 10\left(2\,\|\Sigma'\|_{\max}\,\|\Delta\Sigma'\|_{\max} + \|\Delta\Sigma'\|_{\max}^2\right)\|\Sigma'\|_{\max}\,. \tag{118}$$

Thus,

$$\|\Delta\Sigma\|_{\max} \leq \left(1 + 20\,\|\Sigma'\|_{\max}^2\right)\|\Delta\Sigma'\|_{\max} + 10\,\|\Sigma'\|_{\max}\,\|\Delta\Sigma'\|_{\max}^2\,, \tag{119}$$

and propagating the covariance matrix through the attention layer is locally Lipschitz with a T-independent Lipschitz constant by Eq. (55).

As for the NTK, by the recursion relation in Eq. (57), the propagated term involving $A_1\Theta' A_2^\top$ is handled exactly as in the covariance case, with Σ' replaced by Θ' in the integrand, while the score-law perturbation still enters only through ΔC. The perturbation

$$\mathbb{E}_{\tilde{\boldsymbol{s}}\sim\mathcal{N}(0,C+\Delta C)}\left[\sum_{c,e=1}^{T}\boldsymbol{a}(\boldsymbol{s}_1)_c(\Theta'_{ce}+\Delta\Theta'_{ce})\boldsymbol{a}(\boldsymbol{s}_2)_e\right] \tag{120}$$

can thus be handled T-independently, applying Lemmas B.4, B.5, and B.6, as they only rely on the bounded max-norm of the input kernel.

For the first trace term, F (Eq. (102)), applying Lemmas B.7, B.8, and B.9 together with Lemma B.2 provides us with a T-independent bound on the discrepancy $|\Delta F|$ in Eq. (104) with a Lipschitz constant solely dependent on $\|\Sigma'\|_{\max}$. The second trace term G (Eq. (103)) can be pursued in a similar fashion. Here, the Lipschitz constant will depend on both $\|\Sigma'\|_{\max}$ and $\|\Theta'\|_{\max}$ and will be T-independent nonetheless. Now that we have bounded trace terms, we need to consider the discrepancies of their associated full expressions in Eq. (57), namely,

$$|\Delta\left((2\Sigma'_{ab}+\Theta'_{ab})F\right)| \leq |2\Delta\Sigma'_{ab}+\Delta\Theta'_{ab}|\,|F+\Delta F| + |2\Sigma'_{ab}+\Theta'_{ab}|\,|\Delta F|\,, \tag{121}$$

$$|\Delta(\Sigma'_{ab}G)| \leq |\Delta\Sigma'_{ab}|\,|G+\Delta G| + |\Sigma'_{ab}|\,|\Delta G|\,. \tag{122}$$

Since F and G are uniformly bounded by the trace estimate coming from Eq. (66),

$$|F| \leq 4\,\|\Sigma'\|_{\max}^2\,, \quad |G| \leq 4\,\|\Sigma'\|_{\max}\,\|\Theta'\|_{\max}\,. \tag{123}$$

Since ΔF and ΔG satisfy T-independent local Lipschitz estimates in $\|\Delta\Sigma'\|_{\max}$ and $\|\Delta\Theta'\|_{\max}$, this gives a T-independent local Lipschitz constant in both perturbations in Eqs. (121) and (122).

The element-wise update rule for the covariance matrix through the MLP layer is given in Eq. (62). Because there is no summation or mixing across the T sequence indices, a perturbation $\|\Delta\Sigma\|_{\max}$ merely perturbs k of Eq. (63) by at most $\|\Delta\Sigma\|_{\max}$. For standard activation functions bounded by linear growth, such as ReLU or GeLU, the dual activation integral is locally Lipschitz in the input covariance entries on the nondegenerate covariance domain considered here (Schoenholz et al., 2017). Consequently, the local error scales by a bounded and strictly architecture-dependent constant. For the NTK update through the MLP, Eq. (64) takes the form

$$\Theta_{ab}^{\mathrm{out}} = \Sigma_{ab}^{\mathrm{out}} + \Theta_{ab}\dot{\Sigma}_{ab}^{\mathrm{out}}, \tag{124}$$

where $\dot{\Sigma}_{ab}^{\mathrm{out}}$ denotes the normalized derivative dual kernel. Therefore,

$$\left|\Delta\Theta_{ab}^{\mathrm{out}}\right| \leq \left|\Delta\Sigma_{ab}^{\mathrm{out}}\right| + |\Delta\Theta_{ab}|\left|\dot{\Sigma}_{ab}^{\mathrm{out}} + \Delta\dot{\Sigma}_{ab}^{\mathrm{out}}\right| + |\Theta_{ab}|\left|\Delta\dot{\Sigma}_{ab}^{\mathrm{out}}\right|. \tag{125}$$

The derivative dual kernel is bounded and locally Lipschitz in the 2×2 covariance k, with T-independent constants. This again gives a T-independent local Lipschitz constant in both $\|\Delta\Sigma'\|_{\max}$ and $\|\Delta\Theta'\|_{\max}$.

Since both the attention and MLP kernel recursions possess T-independent max-norm Lipschitz constants, an initial MC error will magnify by at most $(\ell_{\mathrm{Att}}\ell_{\mathrm{MLP}})^L$, for the Lipschitz constants ℓ_{Att} and ℓ_{MLP} of the attention and MLP layers (absorbing LayerNorm), across L transformer blocks. For a finite-depth network, this is a fixed constant factor. In particular, the number of MC samples, N_{MC}, may need to scale exponentially in L in the worst case to maintain the same level of MC error, but this scaling is agnostic of T. □

C Convergence to NNGP

Here, we provide analytic support to Assumption 4.9. This complements existing results (Huang & Yau, 2020; Huang et al., 2021), which assume a fixed input dimension. We proceed by adopting a Bayesian viewpoint and, for simplicity, focus on predicting the T-th token

$$f(X) \equiv \boldsymbol{z}_T^{\mathrm{out}}(X), \quad X \in \mathbb{R}^{T\times d}.. \tag{126}$$

Given an input dataset $\mathcal{D} = \{(X_\nu, y_\nu)\}_{\nu=1}^P$, we let K denote the T-th output kernel matrix

$$K_{\mu\nu} = K(X_\mu, X_\nu) \equiv \Sigma_{TT}^{\mathrm{out}}(X_\mu, X_\nu).. \tag{127}$$

By Cho & Saul (2009, Theorem 3), in the infinite-width limit, $f(X) \sim \mathcal{GP}(0, K)$, which results in the NNGP kernel predictor, for a test point $X_* \in \mathbb{R}^{T\times d}$,

$$f_K(X_*) = \boldsymbol{k}^{*\top}\tilde{K}^{-1}\boldsymbol{y} \tag{128}$$

with a symmetric positive definite (PD)

$$\tilde{K} = K + \kappa I_P, \tag{129}$$

$\kappa > 0$ being the ridge parameter (or observation noise), a vector of covariances

$$\boldsymbol{k}^* = (K(X_*, X_\nu))_{\nu\in[P]} \in \mathbb{R}^P, \tag{130}$$

and $k^* = K(X_*, X_*) \in \mathbb{R}$.

Lemma C.1 (Bounds for $\tilde{K}$)**.** *Under the above setting,*

$$\left\|\tilde{K}^{-1}\right\|_{\max} \le \frac{1}{\kappa}, \qquad \left\|\tilde{K}^{-1}\right\|_2 \le \frac{1}{\kappa}, \qquad \left\|\tilde{K}^{-1}K\tilde{K}^{-1}\right\|_{\max} \le \frac{1}{4\kappa}. \tag{131}$$

Proof. Consider the spectral norm of $\tilde{K}^{-1}$. By definition,

$$\|\tilde{K}^{-1}\|_2 = \sup\left\{\left|\boldsymbol{x}_1^\top\tilde{K}^{-1}\boldsymbol{x}_2\right| \,|\boldsymbol{x}_i \in \mathbb{R}^P : \|\boldsymbol{x}_1\| = 1 = \|\boldsymbol{x}_2\|\right\}. \tag{132}$$

Choosing $\boldsymbol{x}_1 = \boldsymbol{e}_\mu$ and $\boldsymbol{x}_2 = \boldsymbol{e}_\nu$, we get

$$\left|\tilde{K}^{-1}_{\mu\nu}\right| \le \left\|\tilde{K}^{-1}\right\|_2. \tag{133}$$

Using the fact that K is symmetric positive semi-definite (PSD) and $\tilde{K}$ is symmetric PD, we have

$$\mathrm{Spec}\left(\tilde{K}^{-1}\right) = \left\{\frac{1}{\lambda+\kappa}\,\middle|\, \lambda \in \mathrm{Spec}(K)\right\}, \tag{134}$$

so that

$$\left|\tilde{K}^{-1}_{\mu\nu}\right| \le \left\|\tilde{K}^{-1}\right\|_2 = \max_{\lambda\in\mathrm{Spec}(K)} \mathrm{Spec}(\tilde{K}^{-1}) \le \frac{1}{\kappa}, \tag{135}$$

which establishes the two left inequalities. Similarly, we have

$$\mathrm{Spec}(\tilde{K}^{-1}K\tilde{K}^{-1}) = \left\{\frac{\lambda}{(\lambda+\kappa)^2}\,\middle|\, \lambda \in \mathrm{Spec}(K)\right\}. \tag{136}$$

Thus,

$$\left\|\tilde{K}^{-1}K\tilde{K}^{-1}\right\|_2 = \max_{\lambda\in\mathrm{Spec}(K)} \mathrm{Spec}(\tilde{K}^{-1}K\tilde{K}^{-1}) \le \frac{1}{4\kappa}, \tag{137}$$

which establishes the right inequality. □

Proposition 4.10 (NNGP convergence bound)**.** *Consider a finite-width transformer of width N, trained with full-batch gradient descent (GD), weight decay, and additive white Gaussian noise. Let $f(X) = \boldsymbol{z}_T^{\text{out}}(X)$ for $X \in \mathbb{R}^{T\times d}$, and let $f_K(X)$ be its NNGP kernel predictor with $K = \Sigma_{TT}^{\text{out}}$. Then, in standard scaling,*

$$\left|\mathbb{E}_{f\sim P[f]}[f(X_*)] - f_K(X_*)\right| = O\left(\frac{P^\gamma}{N}\right) \tag{16}$$

where $P[f]$ (Naveh et al., 2021, Eq. (3)) is the posterior on the trained f. (Proof in App. C.)

Proof. Naveh et al. (2021) studied the discrepancy between a deep neural network (DNN), f, trained with full-batch gradient descent (GD), weight decay, and additive white Gaussian noise, and its NNGP kernel predictor f_K for the same training set $\mathcal{D}$. At a test point X_*, Naveh et al. (2021) have

$$\mathbb{E}_{f\sim P[f]}[f(X_*)] - f_K(X_*) = \frac{1}{N} f_U(X_*) + O\left(\frac{1}{N^2}\right), \tag{138}$$

where $P[f]$ is the posterior on the trained f. While this work focused on FCNs and convolutional neural networks (CNNs), its results apply to transformers, provided we focus on single-token predictions and, similarly to all works that discuss kernel limits of classification tasks, replace the cross-entropy loss on the logits with an MSE loss w.r.t. one-hot encoding of the categorical labels (Lee et al., 2018). Following this, one again has a Gaussian posterior at infinite width for any specific next-token logit and an Edgeworth expansion controlled by $1/N$ and the fourth cumulant for the leading non-Gaussian corrections. The results we cite below rely solely on this structure.[3] The leading order correction term, $f_U(X_*)$, is given in Naveh et al. (2021, Eq. (11)) by

$$f_U(X_*) = \frac{1}{6}\sum_{\mu_1\mu_2\mu_3=1}^{P} \tilde{U}^*_{\mu_1\mu_2\mu_3}\left(\tilde{y}_{\mu_1}\tilde{y}_{\mu_2}\tilde{y}_{\mu_3} - 3\tilde{K}^{-1}_{\mu_1\mu_2}\tilde{y}_{\mu_3}\right), \tag{139}$$

where U is the 4th-order functional cumulant of the output (Naveh et al., 2021, Eq. (9)) and

$$\tilde{y}_{\mu_i} = \sum_{\nu=1}^{P}\tilde{K}^{-1}_{\mu_i\nu}y_\nu, \quad \tilde{U}^*_{\mu_1\mu_2\mu_3} = U^*_{\mu_1\mu_2\mu_3} - \sum_{\nu,\nu'=1}^{P} U_{\mu_1\mu_2\mu_3\nu}\tilde{K}^{-1}_{\nu\nu'}\boldsymbol{k}^*_{\nu'}. \tag{140}$$

The asterisk on the right-hand side of Eq. (140) indicates evaluation, namely, $\boldsymbol{k}^*_{\nu'} = K(X_*, X_{\nu'})$ and $U^*_{\mu_1\mu_2\mu_3} = U\left(X_*, X_{\mu_1}, X_{\mu_2}, X_{\mu_3}\right)$ and y_ν is either a regression target or an indicator function for the logit taking values of $0, 1$ (or with bias-free one-hot encoding, $-1/V, 1-1/V$ where V is the size of the vocabulary).

The scale of $f_U(X_*)$ as a function of P determines the scale of the discrepancy between the NNGP kernel predictor and the finite DNN. We will therefore bound the basic ingredients of Eq. (139) independently of T. First, by definition, we have $y_\mu = O(1)$. We also have $\tilde{K}^{-1}_{\mu\nu} = O(1)$ by Lemma C.1. Then,

$$\|\tilde{\boldsymbol{y}}\|_2 \le \left\|\tilde{K}^{-1}\right\|_2 \|\boldsymbol{y}\|_2 \le \frac{\|\boldsymbol{y}\|_2}{\kappa} = O(\sqrt{P}), \tag{141}$$

applying Lemma C.1 again. Second, the output $f(X)$ has an $O(1)$ Gaussian tail at initialization, and its variance is bounded irrespective of T, yielding $\boldsymbol{k}^*_\mu = O(1)$. Similarly, the 4th-order cumulant between any four points, including a test point, also obeys $U_{\mu_1\ldots\mu_4} = O(1)$ and $U^*_{\mu_1\ldots\mu_3} = O(1)$. A crude estimate would, thus, yield $\tilde{U}^*_{\mu_1\mu_2\mu_3} = O(P^2)$. However, we will show $\tilde{U}^*_{\mu_1\mu_2\mu_3} = O(\sqrt{P})$.

By definition, for a function u in the reproducing kernel Hilbert space (RKHS), $\mathcal{H}_K$, of K, we have

$$u(X) = \langle K(X,\cdot), u\rangle_{\mathcal{H}_K}, \tag{142}$$

and, in particular, $\langle K(X,\cdot), K(X',\cdot)\rangle_{\mathcal{H}_K} = K(X,X')$. Consequently,

$$|u(X)| \le \|u\|_{\mathcal{H}_K}\|K(X,\cdot)\|_{\mathcal{H}_K} = \|u\|_{\mathcal{H}_K}\sqrt{K(X,X)}. \tag{143}$$

[3] As in the original work, these results do not apply for mean-field scaling where the theory at infinite N stops being Gaussian.

Next, taking

$$u_K(X) = \sum_{\nu,\nu'=1}^{P} U_{\mu_1\mu_2\mu_3\nu} \tilde{K}^{-1}_{\nu\nu'} K(X, X_{\nu'}), \tag{144}$$

we note that $u_K(X_*) = U^*_{\mu_1\mu_2\mu_3} - \tilde{U}^*_{\mu_1\mu_2\mu_3}$. Its $\mathcal{H}_K$-norm obeys

$$\begin{aligned}
\|u_K\|^2_{\mathcal{H}_K} &= \left\langle \sum_{\mu,\mu'=1}^{P} U_{\mu_1\mu_2\mu_3\mu} \tilde{K}^{-1}_{\mu\mu'} K(\cdot, X_{\mu'}), \sum_{\nu,\nu'=1}^{P} U_{\mu_1\mu_2\mu_3\nu} \tilde{K}^{-1}_{\nu\nu'} K(\cdot, X_{\nu'}) \right\rangle_{\mathcal{H}_K} \\
&= \sum_{\mu,\mu'=1}^{P} \sum_{\nu,\nu'=1}^{P} U_{\mu_1\mu_2\mu_3\mu} \tilde{K}^{-1}_{\mu\mu'} \langle K(\cdot, X_{\mu'}), K(\cdot, X_{\nu'}) \rangle_{\mathcal{H}_K} U_{\mu_1\mu_2\mu_3\nu} \tilde{K}^{-1}_{\nu\nu'} \\
&= \sum_{\mu,\mu'=1}^{P} \sum_{\nu,\nu'=1}^{P} U_{\mu_1\mu_2\mu_3\mu} \tilde{K}^{-1}_{\mu\mu'} K(X_{\mu'}, X_{\nu'}) \tilde{K}^{-1}_{\nu'\nu} U_{\mu_1\mu_2\mu_3\nu} \\
&= \sum_{\mu\nu=1}^{P} U_{\mu_1\mu_2\mu_3\mu} \left(\tilde{K}^{-1} K \tilde{K}^{-1} \right)_{\mu\nu} U_{\mu_1\mu_2\mu_3\nu} \\
&\leq \frac{\|\boldsymbol{u}\|_2^2}{4\kappa}
\end{aligned} \tag{145}$$

where, in the final inequality, we applied Lemma C.1, with $\boldsymbol{u} = (U(X_{\mu_1}, X_{\mu_2}, X_{\mu_3}, X_\nu))_{\nu\in[P]} \in \mathbb{R}^P$. Applying Eq. (143), we obtain

$$|u_K(X_*)| < \frac{\|\boldsymbol{u}\|_2}{\sqrt{4\kappa}} \sqrt{K(X_*, X_*)} = O(\sqrt{P}), \tag{146}$$

where we assumed properly normalized kernels and target functions. Consequently, we obtain $\tilde{U}^*_{\mu_1\mu_2\mu_3} = O(\sqrt{P})$, as stated, and, putting all bounds together,

$$\left| \mathbb{E}_{f\sim P[f]}[f(X_*)] - f_K(X_*) \right| \leq \frac{CP^\gamma}{N} + \frac{C'}{N^2}, \tag{147}$$

for $\gamma = 5$. □

D THE FULLY CONNECTED NETWORK

Consider a fully connected network (FCN) acting on (a flattened) $X \in \mathbb{R}^{T\times d}$. At the l-th layer, $l \geq 2$, the network computes

$$z^l_i(X) = b^l_i + \sum_{j=1}^{N_{l-1}} W^l_{ij} a^{l-1}_j(X), \quad a^{l-1}_j(X) = \phi\left(z^{l-1}_j(X)\right), \quad i \in [N_l], j \in [N_{l-1}], \tag{148}$$

with weights $\boldsymbol{b}^l \in \mathbb{R}^{N_l}$ and $W^l \in \mathbb{R}^{N_l \times N_{l-1}}$ drawn via

$$b^l_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2_{\boldsymbol{b}}), \qquad W^l_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{\sigma^2_W}{N_{l-1}}\right). \tag{149}$$

By our convention $a^0(X) = \text{vec}(X) \in \mathbb{R}^{Td}$ so $N_0 = Td$. As $N_l \to \infty$, the preactivations converge in distribution to a Gaussian process $z^l_i \sim \mathcal{GP}(0, K^l)$. The complexity of evaluating the FCN NNGP kernel (Lee et al., 2018) for a sequence of length T is dominated by the dot product scaling as $O(Td)$. By Lee et al. (2018), the recursion is initialized with

$$K^1(X, X') = \mathbb{E}\left[z^1_j(X) z^1_j(X')\right] = \sigma^2_{\boldsymbol{b}} + \sigma^2_W \left(\frac{\text{vec}(X) \cdot \text{vec}(X')}{dT} \right), \tag{150}$$

with the recursion relation, for $l \geq 2$,

$$K^l(X, X') = \sigma^2_{\boldsymbol{b}} + \sigma^2_W F_\phi\left(K^{l-1}(X, X'), K^{l-1}(X, X), K^{l-1}(X', X')\right), \tag{151}$$

given via a deterministic function F_ϕ whose form depends only on the nonlinearity ϕ. Crucially, the NNGP kernel recursions are element-wise on the Gram matrix, involving no mixing across sequence indices. Consequently, depth (L) only adds an additive $+L$ term to the inference-time computational complexity. Due to various fixed points which appear in these recursions, the infinite-depth limit is also likely to be of finite computational complexity (Schoenholz et al., 2017).

E Detailed Experiments

E.1 Combinatorial Tasks

Here, we provide a detailed description of each combinatorial task used in our experiments (Sec. 3.4). We provide a symbolic formulation for each task following Sec. 3.1. Note that different choices of score function ψ_X may lead to the same underlying prediction task. Our choice of ψ_X is intended to formalize a natural notion of correctness or optimality induced by the task instance X; throughout, we use task-structural choices rather than arbitrary post hoc encodings. As combinatorial optimization problems, SPP and MinCut are naturally combinatorial tasks with configuration sets being the set of feasible solutions and score function being the objective per task instance. In our experiments, we use those two combinatorial tasks in value form, predicting the value of the optimal solution.

Induction. We consider a vocabulary of size 1024 with sequences of length T. For each sequence, we first generate random tokens and select a 'trigger' token from the vocabulary. To prevent ambiguity, we strictly enforce that the trigger appears only where intended by replacing any pre-existing random occurrences with the subsequent token in the vocabulary (modulo 1024). We then inject the trigger token at a random index $i_K \in \{0, \dots, \lceil T/2 \rceil - 1\}$ and again at the final sequence index $T-1$. As a symbolic combinatorial task $g : \mathcal{X} \to \mathcal{Y}$ in solution form, we take $\mathcal{X}$ to be all legal finite sequences in the vocabulary and $\mathcal{Y}$ to be the vocabulary set. For each sequence $X \in \mathcal{X}$, the configuration set $\mathrm{Conf}(X)$ consists of all tokens of the vocabulary. The score function $\psi_X : \mathrm{Conf}(X) \to \mathbb{R}$ can be chosen to be $\psi_X(\rho) = \mathbf{1}_{X_{i_K+1} \neq \rho}$. The tokens are then mapped to learned embeddings ($X \in \mathbb{R}^{T \times d}$) and augmented with rotary positional embeddings (RoPE) (Su et al., 2021). The objective is to predict the token following the final trigger. To solve this, the transformer must perform an induction head operation (Elhage et al., 2021): it must locate the previous occurrence of the trigger token (at index i_K) and copy the token immediately following it ($\boldsymbol{x}_{i_K+1}$).

Sorting Vocabulary. We consider a vocabulary of integers $\mathcal{V} = \{0, \dots, V-1\}$ with vocabulary size $V \in \{100, 200, 300\}$. For each instance, we generate a sequence $u = (u_1, \dots, u_T)$ of length T, where elements are sampled uniformly with replacement from $\mathcal{V}$, and let $s = (s_1, \dots, s_T)$ denote a nondecreasing rearrangement of u. As a symbolic combinatorial task $g : \mathcal{X} \to \mathcal{Y}$ in solution form, we let $\mathcal{X} = \mathcal{Y}$ be the set of all finite sequences in the vocabulary. For each $u \in \mathcal{X}$ the configuration set $\mathrm{Conf}(u)$ is the set of all permutations ρ of u and $\psi_u(\rho) = \sum_{a=1}^{T-1} (\rho_a - \rho_{a+1})_+$, where $(x)_+ = \max\{x, 0\}$. During training, we construct the full sequence under teacher forcing by concatenating the unsorted list u, a special separator token (SEP), and the sorted s. The inputs are mapped to learned embeddings and augmented with RoPE. The model is trained using a standard causal next-token prediction objective over the concatenated sequence $[u, \mathrm{SEP}, s]$. However, we apply a mask to the loss function to ignore predictions on the unsorted prefix u, calculating the loss only on the generation of the separator and the sorted suffix s. Accuracy is evaluated at the token level over the masked sorted suffix. At inference, this task requires the model to generate the sorted sequence autoregressively from the unsorted prefix: at each step of the generation phase, it is naturally interpreted as needing to attend back to the unsorted prefix u to identify and copy the next smallest element that has not been generated yet.

String Matching. We consider random sequences of length T generated from a vocabulary of size 26. Let M be a specific pattern of length 3 in the vocabulary. The objective is to determine whether M appears anywhere within the sequence. As a symbolic combinatorial task $g : \mathcal{X} \to \mathcal{Y}$ in value form, we let $\mathcal{X}$ be the set of all finite sequences in the vocabulary and $\mathcal{Y} = \{0, 1\}$. For each $X \in \mathcal{X}$, $\mathrm{Conf}(X) = \{X[m : m+2]\}_{m=0}^{T-3}$ and $\psi_X(\rho) = \mathbf{1}_{\rho \neq M}$. Then

$$g(X) = \min_{\rho \in \mathrm{Conf}(X)} \psi_X(\rho) = \begin{cases} 0 & M \subseteq X \\ 1 & M \not\subseteq X \end{cases}. \tag{152}$$

We formulate the learning problem as binary classification, with the model trained to predict the label $1 - g(X)$. The input for each task instance is constructed by concatenating the random sequence X, a special separator token (SEP), and the target pattern M. To prevent the model from learning trivial heuristics, we construct adversarial negative examples by ensuring the negative samples contain near-misses (e.g., matching 2 characters of the pattern).

Context-free Grammar (CFG) Recognition Problem. A context-free grammar (CFG) (Sipser, 2006, Definition 2.2) is a 4-tuple $G = (V, \Sigma, R, S)$, where V is a finite set called the variables, Σ is a

finite set, disjoint from V, called the terminals, R is a finite set of substitution rules, with each rule being a variable and a string of variables and terminals, and $S \in V$ is the start variable. The language $L(G)$ of G consists of all terminal strings derivable from S via R. A parse tree of $w \in L(G)$ is one such legal derivation. We evaluate the model's ability to recognize strings in the language $L(G)$.

We define a strict Chomsky Normal Form (CNF) grammar with $|V| = 20$ variables and $|\Sigma| = 16$ terminals. The branching rules consist of binary non-terminal productions ($A \to BC$) and terminal productions ($A \to a$). To maintain criticality and ensure the average derivation tree depth scales as $O(\sqrt{T})$, the binary branching weights are normalized by the spectral radius of their transition matrix (Chi, 1999).

Crucially, our generation procedure allows us to sample strings of any arbitrary exact length T. While the grammar strictly enforces binary branching ($1 \to 2$) at every internal node, the derivation trees are not forced to be perfectly balanced. A parent node A generating a total sequence of length l can split asymmetrically into a left child B that recursively generates a sub-tree of length k, and a right child C that recursively generates a sub-tree of length $l - k$ (for any $1 \leq k < l$). Both B and C will continue to strictly apply binary branching rules internally until they hit their respective target lengths at the terminal leaves.

The dataset we generated was exactly balanced, consisting of 50% strings in $L(G)$ (positive examples) and 50% negative examples. To ensure the model learns robust parsing rules rather than surface-level statistics, negative examples were generated via an adversarial substitution procedure: we first sample a valid parse tree of length T, randomly select an intermediate length scale (to avoid power-law length bias), and pick an internal node A at that scale. We then completely replace A's sub-tree with a new valid sub-tree of the exact same length, but generated from an adversarial variable whose required parent production rule does not exist in the grammar.

Random Geometric Graphs. We test algorithmic capture on two types of combinatorial graph problems. To that end, we generate random (undirected) graph instances following the Random Geometric Graph (RGG) setup (Penrose, 2003). First, we draw T points $\boldsymbol{x}_a$, $a = 1, \ldots, T$, i.i.d. from a uniform measure $\mu_{\boldsymbol{x}}$ on $[-1, 1]^d$. We consider each $\boldsymbol{x}_a$ as a node in our sampled RGG. The nodes $\boldsymbol{x}_1$ and $\boldsymbol{x}_T$ take the special role of source and target, respectively. Next, we fix $r \geq 0$ and add an edge of weight (or capacity) 1 between any two nodes $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ with $\|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2 \leq r$. Unless stated otherwise, we parametrize r via the expected degree α using Dall & Christensen (2002, Eq. (7)) and choose α near the regime where a giant linear-sized component emerges. We denote our joint input measure by $\mu_{X,T} := \mu_{\boldsymbol{x}}^{\otimes T}$. From an average-time or heuristic complexity perspective (Levin, 1986; Impagliazzo, 1995), this distribution has low sampling complexity but generates relatively hard graph instances.

Shortest Path Problem (SPP). Consider a finite undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V}$ of size $V = |\mathcal{V}|$, edge set $\mathcal{E}$ of size $E = |\mathcal{E}|$, and two distinguished nodes: a source v_s and target v_t. A (node) path in $\mathcal{G}$ from source to target is a sequence $\pi = (v_0, \ldots, v_n)$ of nodes such that $\{v_{i-1}, v_i\} \in \mathcal{E}$ for $i \in [n]$ beginning with $v_0 = v_s$ and ending with $v_n = v_t$. Graph weighting constitutes a map $c : \mathcal{E} \to \mathbb{R}_{\geq 0}$. In our RGG setting, weighting is constant $c : \mathcal{E} \to \{1\}$. The SPP consists of finding an optimal path π^* for

$$\pi^* \in \underset{\substack{\pi=(v_0,\ldots,v_n) \\ v_0=v_s, v_n=v_t}}{\operatorname{argmin}} \sum_{i=1}^{n} c\left(\{v_{i-1}, v_i\}\right). \tag{153}$$

For our $\mu_{X,T}$, we consider the RGG at the critical connectivity threshold. The average-case time complexity for source-target SPP is $\mathbb{E}_{X \sim \mu_{X,T}}[t(X)] = \mathbb{E}_{X \sim \mu_{X,T}}[E(X) + V(X)] = O(T \log(T))$, matching standard worst-case complexity for the single-pair SPP on an undirected unit-weight graph using Breadth-First Search (BFS) and critical connectivity expectation $\mathbb{E}_{X \sim \mu_{X,T}}[E(X)] = \Theta(T \log T)$. As noted in Sec. 3, this places the distributional SPP task in the EPTHS class of complexity $O(T^{1+\epsilon})$ under $\mu_{X,T}$ for every $\epsilon > 0$.

MinCut / MaxFlow. Starting from the RGG and graph setup above, we adapt it to add directions to edges, i.e., consider $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. In a MinCut / MaxFlow network (Bertsekas, 1998, Proposition 3.2), nodes, apart from the source and target, are termed relay nodes. In our directed graph adaptation, relay nodes become bidirectional (with equal capacity), all edges incident to the source have v_s as their initial node, and all edges incident to the target have v_t as their terminal node as in Ramamoorthy

et al. (2005, Definition 2).[4] A cut $\mathcal{S} \subset \mathcal{V}$ in the graph is a subset containing the source $v_s \in \mathcal{S}$ whose complement $\overline{\mathcal{S}} = \mathcal{V} \setminus \mathcal{S}$ contains the target $v_t \in \overline{\mathcal{S}}$. The MinCut problem consists of finding an optimal cut $\mathcal{S}^*$ for

$$\mathcal{S}^* \in \operatorname*{argmin}_{\substack{\mathcal{S}\subset\mathcal{V} \\ v_s\in\mathcal{S} \\ v_t\in\overline{\mathcal{S}}}} \sum_{\substack{(v,\bar{v})\in\mathcal{E} \\ v\in\mathcal{S},\bar{v}\in\overline{\mathcal{S}}}} c\,(v,\bar{v})\,. \tag{154}$$

Average-case complexity for MinCut/MaxFlow is $O(\mathbb{E}[VE])$, matching standard worst-case complexity for MaxFlow (Orlin, 2013). For similar considerations as in SPP, this places MinCut/MaxFlow in the EPTHS class of complexity $O(T^{2+\epsilon})$ under $\mu_{X,T}$ for every $\epsilon > 0$.

E.2 Engineering

We trained all models either on an Nvidia RTX 4080 or an Nvidia RTX 4090 GPU using a standard decoder-only transformer architecture (with the exception of CFG and SPP, which use a bidirectional encoder) equipped with GeLU activations. The estimated compute per experiment was ~ 24 hours.

For the four capture tasks Induction, Sorting Vocabulary, String Matching, and CFG, we utilized an *incremental curriculum* for evaluating adaptation costs. Instead of training the model from scratch for each extrapolation horizon T, we pre-trained the model at a base size T_0 to a specific accuracy threshold δ. We then iteratively adapted the model to increasing horizons, dynamically sampling sequences of length $T_{\text{curr}} \sim \mathcal{U}[T_{\text{prev}}, T_{\max}]$, where $T_{\max}$ was typically increased using a 20% multiplicative schedule. The reported adaptation budget P represents the *cumulative sum* of samples required to maintain the δ accuracy threshold across all incremental steps. The training configurations were as follows:

- **Induction**: Causal Transformer, layers $L = 2$, $d = 256$, heads $H = 4$. We used Rotary Positional Embeddings (RoPE) with an increased base frequency ($\rho = 500{,}000$) to facilitate stable extrapolation up to $T_{\max} = 10{,}000$. The pre-training base was $T_0 = 50$. Averaged over 5 seeds.
- **Sorting Vocabulary**: Causal Transformer, $L = 2$, $d = 128$, $H = 2$, MLP dimension 1024. We used RoPE with standard base 10,000. The evaluated target threshold was $\delta = 5\%$ token-level error. The pre-training base was $T_0 = 50$. Averaged over 3 seeds per vocabulary size.
- **String Matching**: Causal Transformer, $L = 3$, $d = 64$, $H = 1$, MLP dimension 256. We used RoPE with base 100,000. Models were evaluated at thresholds $\delta \in \{10\%, 5\%, 2\%\}$. The pre-training base was $T_0 = 50$. Averaged over 5 seeds.
- **CFG**: Bidirectional Transformer with mean-pooling, $L = 12$, $d = 128$, $H = 4$. Models were evaluated at thresholds $\delta \in \{40\%, 20\%, 10\%\}$. The pre-training base was $T_0 = 10$. Averaged over 4 seeds.

For the combinatorial optimization tasks (SPP and MinCut), models were trained using the standard protocol on the whole sequence interval rather than the incremental protocol described above. The configurations were:

- **Shallow SPP**: $L = 4, d = 128, B = 1024$.
- **Deep SPP**: $L = 40, d = 128, B = 1024$.
- **Shallow MinCut**: $L = 4, d = 128, B = 1024$.
- **Deep MinCut**: $L = 40, d = 128, B = 1024$.

The pre-training base was $T_0 = 10$ in either case.

[4] Equivalently, we may take the complete directed edge set $\mathcal{E} = \mathcal{V} \times \mathcal{V}$ (optionally excluding self-loops) and extend edge capacity c by setting $c(u, v) = 0$ for all directed pairs $(u, v) \in \mathcal{E}$ not present in the geometric graph; this leaves the MinCut/MaxFlow value unchanged.