Compact Proofs of Model Performance via Mechanistic Interpretability

Jason Gross [1] **Rajashree Agrawal** [1] **Thomas Kwa** [*1] **Euan Ong** [*2] **Chun Hei Yip** [*2] **Alex Gibson** [+2] **Soufiane Noubir** [+2] **Lawrence Chan** [1]

Abstract

We propose using mechanistic interpretability – techniques for reverse engineering model weights into human-interpretable algorithms – to derive and compactly prove formal guarantees on model performance. We prototype this approach by formally proving lower bounds on the accuracy of 151 small transformers trained on a Max-of-K task. We create 102 different computer-assisted proof strategies and assess their length and tightness of bound on each of our models. Using quantitative metrics, we find that shorter proofs seem to require and provide more mechanistic understanding. Moreover, we find that more faithful mechanistic understanding leads to tighter performance bounds. We confirm these connections by qualitatively examining a subset of our proofs. Finally, we identify compounding structureless noise as a key challenge for using mechanistic interpretability to generate compact proofs on model performance.

1. Introduction

One approach to ensuring the safety and reliability of powerful AI systems is via formally verified proofs of model performance (Seshia et al.; Dalrymple et al., 2024). If we hope to deploy formal verification on increasingly large models (Hoffmann et al., 2022; Kaplan et al., 2020) with powerful emergent capabilities (Wei et al., 2022) across more diverse and broader domains (Brown et al., 2020; Reed et al., 2022), we will need *compact* proofs of generalization properties on *specific* models that certify *global* robustness. However, existing approaches tend to use proof strategies that suffer from bad asymptotic complexity, while verifying either generalization properties of training procedures or local robustness properties of specific models.

One key challenge to verification is that neural network architectures are highly expressive (Szegedy et al., 2013; Zhang et al., 2021), and models with similar training procedure and performance may still have learned significantly different weights (Nanda et al., 2023; Chughtai et al., 2023). This expressivity makes it difficult to adequately *compress* explanations of global model behavior in ways that *correspond* closely enough to the model's actual mechanisms to be useful for efficient verification without being too *lossy*, especially when using only knowledge of the architecture or training procedure. We propose verifying model performance using understanding derived from *mechanistic interpretability* (Section 2) – that is, reverse engineering the specific implementation of the algorithm from the learned weights of particular models. Knowledge of the specific implementation allows us to construct less lossy simplifications of the model, and more efficiently reason about model performance over possible inputs.

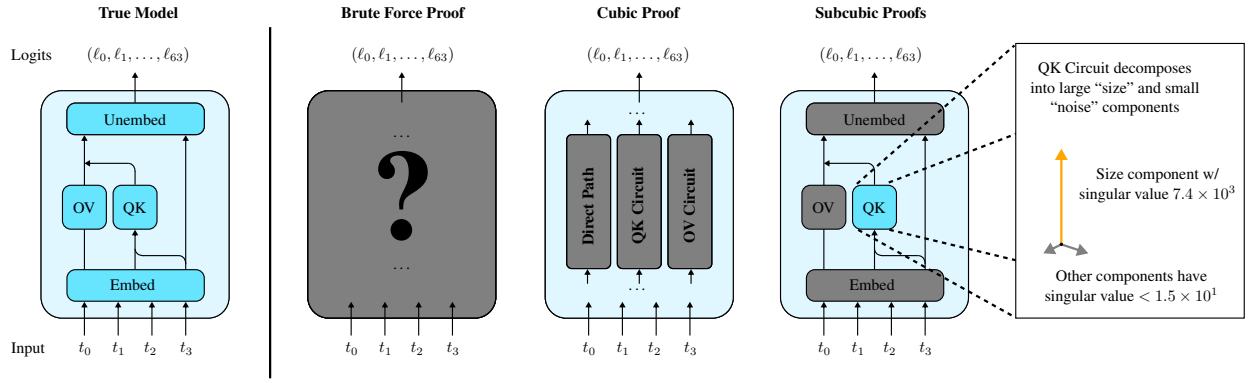
In this work, we provide a case study of translating mechanistic interpretations into compact proofs. We train an attention-only transformer on a Max-of-K task with 151 random seeds (Section 3), and then reverse engineer the models using standard mechanistic interpretability techniques. We use our understanding to define a set of 102 different computer-assisted proof strategies with varying tightness of bound and with different asymptotic complexity and number of required floating-point operations (Section 4).[1]

We define a quantitative metric to assess the mechanistic understanding used in a proof strategy by the dimensionality of the function space that the proof strategy must consider, which we deem the *unexplained dimensionality* of the proof strategy (Sections 5.1 and A.2). Using this metric, we find a negative relationship between proof length and degree of understanding. We qualitatively examine proof strategies to confirm and explain this relationship, finding that more compact proofs both require and provide more mechanistic understanding. We also find suggestive evidence that the trade-off between proof length and tightness of bound is modulated by the faithfulness of the mechanistic under-

[*]Equal contribution [+]Equal contribution [1]Independent [2]University of Cambridge. Correspondence to: Jason Gross <jasongross9@gmail.com>.

Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]Our 102 proof strategies are can be broken up as $1 + 1 + 10 \times 5 \times 2$: two standalone strategies, and a class of strategies parameterized on three axes of cardinality 10, 5, and 2 (Appendix H).



	True Model	Brute Force Proof	Cubic Proof	Subcubic Proofs
FLOPs Required:		8.76×10^{11}	3.52×10^7	4.68×10^6
Accuracy Lower Bound:		99.92 %	98.45 %	64.26 %
Unexplained Dimension:		1.07×10^9	1.28×10^4	4.42×10^3
Asymptotic Complexity:		$\mathcal{O}({d_{\text{vocab}}}^{n_{\text{ctx}}})$	$\mathcal{O}({d_{\text{vocab}}}^3 \cdot n_{\text{ctx}})$	$\mathcal{O}(d_{\text{vocab}} \cdot {d_{\text{model}}}^2 \cdot n_{\text{ctx}})$

Figure 1: We construct proofs using different degrees of mechanistic interpretation. (Left) The models we consider in this paper are one-layer attention-only transformers, and so contain three "paths": the OV circuit, the QK circuit, and the direct path. (Right) For the brute-force proof (Section 4.3.1), we treat the model as a black box and thus need to check all possible combinations of inputs. For the cubic proof (Section 4.3.1), we decompose the model into its three corresponding paths, but still check the correctness of each path via brute force. Finally, in some subcubic proofs (Section 4.3), we use all parts of the mechanistic interpretation presented in Section 3. (Bottom) For each of the three categories of proof, we report the number of FLOPs used in computing the certificate (lower=better, Appendix A.3), lower bound on model accuracy (higher=better), effective dimension of the unexplained parts of the model (lower=better, Appendix A.2), and asymptotic complexity of the proof strategy as we scale the inputs and model (lower=better). Significantly more compact proofs have vacuous accuracy bounds by default. Using more mechanistic understanding allows us to recover some, but not all, of the accuracy bounds on these more compact proofs, as our understanding is not fully faithful to the model internals.

standing used to derive the proof (Section 5.2).[2]

However, we also identify compounding structureless noise as a key challenge for generating compact proofs on model behavior (Sections 5.3 and G.2.5). The implementation of algorithms inside of neural networks may contain components that defy mechanistic understanding and appear to us as "noise". When we don't know how noise composes across model components, establishing a bound requires pessimizing over the ways the composition could occur. Worst-case noise can quickly grow even when the empirical noise is small, leading to vacuous performance bounds.

2. Mechanistic interpretability for proofs

Generalization bounds on global performance In the style of prior mechanistic interpretability evaluation work (Chan et al., 2022), we target theorem templates that establish bounds on the expected global performance of the model. Let $\mathcal{M} : X \to Y$ be a model (here assumed to be a neural network), $\mathcal{D}$ be a probability distribution over input-label pairs $(l, \mathbf{t}) \in L \times X$, notated as $\mathcal{D}|_X$ when marginalized over labels, and $f : L \times Y \to \mathbb{R}$ be a scoring function for evaluating the performance of the model. Then, we seek to establish lower bounds b on the expected $\bar{s}$ as the form:

$$\bar{s} := \mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}}\left[f(l, \mathcal{M}(\mathbf{t}))\right] \geq b. \tag{1}$$

As f can be any metric, this is a fully general template for theorems that can capture any aspect of model performance for which we have a formal specification. However, in this work we restrict f to be the 0-1 loss and $\mathcal{D}|_X$ to be uniform, so our theorems lower bound the accuracy of the model. Our proof methodology generalizes straightforwardly to other input distributions, and only a little work is required to generalize from accuracy to log-loss (Appendices A.7 and A.5).

Proof template The proofs of model performance in this work have two components: a computational component C : model weights $\to \mathbb{R}$ and a non-computational component Q arguing that for any model $\mathcal{M}'$, $C(\mathcal{M}') \leq \mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}} f(l, \mathcal{M}'(\mathbf{t}))$, thus implying that C generates a valid lower bound for the performance of $\mathcal{M}$. The whole proof is Q paired with a trace of running C that certifies its output on $\mathcal{M}$.[3] Here, $b = C(\mathcal{M})$. As even the size of the model parameters is much larger than any reasonable Q, we approximate the length of a proof pair C, Q by the length of a trace of $C(\mathcal{M})$.

[2]Code for reproducing our results can be found at https://github.com/JasonGross/guarantees-based-mechanistic-interpretability/.

[3]Other components of the proof to account for the difference between floating point numbers and reals are described in Appendix A.4. Note that all proofs explicitly given in this paper are of Q only; we do not include any traces of running C.

Proof compactness vs. tightness of bound Different proof strategies make different tradeoffs between compactness and tightness of bound. For example, consider two extreme proof strategies: We can "prove" a vacuous bound using a null proof. On the other hand, in the brute-force proof, we simply run the model on the entirety of $\mathcal{D}$ to achieve $b = \bar{s}$, albeit with a very long proof.

We quantify the length of $C(\mathcal{M})$ using two metrics: the *asymptotic time complexity* of C as we scale the size of the model and the input $\mathbf{t}$, as well as the empirical average *number of floating point operations* required to evaluate $C(\mathcal{M}')$ over a given set of models $\{\mathcal{M}_i\}$. We measure *tightness of bound* of $C(\mathcal{M})$ using the ratio of the bound to the true accuracy: $b/\bar{s}$.

Proof as pessimal ablation A standard way of assessing the faithfulness of mechanistic interpretability is by ablating the parts of the model that your interpretation does not explain (Wang et al., 2022; Chan et al., 2022; Heimersheim & Nanda, 2024). In this framework, proofs can be thought of as performing a *pessimal ablation* over the unexplained parts of the model – we set the remaining components of the model (the "noise") with values drawn from $\mathcal{D}$ that minimize the performance of the model. However, the number of ablations required for a complete argument might be quite high. Thus, we construct *relaxations* (Appendix A.1) over input sequences, such that performing pessimal ablations on a smaller number of relaxed input sequences is sufficient to lower bound the performance on $\mathcal{D}$.

3. Experimental setting

We study our approach to generating compact proofs in a simple toy setting: one-layer transformers trained to output the max of k integers.

Model Architecture We study one-layer, one-head, attention-only transformers with no biases but with learned positional embeddings, with vocabulary size d_{vocab}, model and head dimension $d = d_{\text{model}} = d_{\text{head}}$, and context length $n_{\text{ctx}} := k$. The model parameters consist of the $n_{\text{ctx}} \times d_{\text{model}}$ positional embedding P; the $d_{\text{vocab}} \times d_{\text{model}}$ token embed E; the $d_{\text{model}} \times d_{\text{model}}$ query, key, value, and output matrices of the attention head Q, K, V, and O; as well as the $d_{\text{model}} \times d_{\text{vocab}}$ unembed matrix U. We assume (as is standard in language modeling) that $d_{\text{model}} < d_{\text{vocab}}$.

For an $n_{\text{ctx}} \times d_{\text{vocab}}$ one-hot encoding $\mathbf{x}$ of an input sequence $\mathbf{t}$, we compute the logits of the model as follows:

$$h^{(0)} = \mathbf{x}E + P \qquad \text{Initial residual stream } (n_{\text{ctx}} \times d_{\text{model}})$$

$$\alpha = h^{(0)} Q K^T {h^{(0)}}^T / \sqrt{d} \qquad \text{Attention matrix } (n_{\text{ctx}} \times n_{\text{ctx}})$$

$$h^{(1)} = \sigma^*(\alpha) \cdot h^{(0)} V O + h^{(0)} \qquad \text{Final residual stream } (n_{\text{ctx}} \times d_{\text{model}})$$

$$\mathcal{M}(\mathbf{t}) = \ell = h^{(1)}_{n_{\text{ctx}}-1} U \qquad \text{Final seq. position logits } (d_{\text{vocab}})$$

where σ^* is the masked softmax function used in causal attention. Because we only look at outputs of the model above the final sequence position $i = n_{\text{ctx}} - 1$, we also denote this position as the "query position" and the value of the token in this position as t_{query}, one-hot encoded as x_{query}. The model's prediction is the token corresponding to the max-valued logit $\ell_{\max}$.

Task Specifically, we study the setting with $n_{\text{ctx}} = k = 4$ because it is the largest sequence length for which we can feasibly evaluate the brute-force proof. We set hidden dimension $d_{\text{model}} = 32$ and a vocabulary of size $d_{\text{vocab}} = 64$ comprising integers between 0 and 63 inclusive. For an input sequence $\mathbf{t} = [t_0, t_1, t_2, t_3]$, we denote the *true* maximum of the sequence by $t_{\max}$. Outputting the correct behavior is equivalent to outputting logits ℓ such that $\Delta\ell_{t^*} := \ell_{t^*} - \ell_{\max} < 0$ for all $t^* \neq t_{\max}$. We trained 151 models on this task. Models achieved an average accuracy of 0.9992 ± 0.0015 over the entire data distribution.

Path decomposition Following prior work (Elhage et al., 2021), we expand the logits of the model and split the paths through the model into three components – the QK circuit, the OV circuit, and the direct path in Equation 2.

Intuitively, the QK circuit determines *which* tokens the model attends to from a particular query token and sequence position, while the OV circuit *processes* the tokens and sequence positions the model attends to. The direct path is simply the skip connection around the attention head.

We further divide the QK and OV circuits into token (position-independent) and position-dependent components. Let $P_{\text{avg}} = \frac{1}{n_{\text{ctx}}} \sum_i P_i$ be the average position embeds across positions (of size d_{model}), and let $\bar{\mathbf{P}}$ denote either $\mathbf{1}_{n_{\text{ctx}}} \otimes P_{\text{avg}}$ or $\mathbf{1}_{d_{\text{vocab}}} \otimes P_{\text{avg}}$ depending on context, the result of broadcasting P_{avg} back into the shape of P or E (that is, $n_{\text{ctx}} \times d_{\text{model}}$ or $d_{\text{vocab}} \times d_{\text{model}}$). Similarly, let $\mathbf{P}_q = \mathbf{1}_{d_{\text{vocab}}} \otimes P_{\text{query}}$ be the result of broadcasting P_{query}. Then for one-hot encoded $\mathbf{x}$, we can rewrite the QK and OV circuits, as well as the direct path, as follows:

$$\text{QK circuit} = x_{\text{query}} \Big(\underbrace{\mathbf{E}_q Q K^T \bar{\mathbf{E}}^T}_{\text{EQKE}} \mathbf{x}^T + \underbrace{\mathbf{E}_q Q K^T \hat{\mathbf{P}}^T}_{\text{EQKP}} \Big)$$

$$\text{OV circuit} = \mathbf{x} \underbrace{\bar{\mathbf{E}} V O U}_{\text{EVOU}} + \underbrace{\hat{\mathbf{P}} V O U}_{\text{PVOU}} \qquad \text{Direct Path} = x_{\text{query}} \underbrace{\mathbf{E}_q U}_{\text{EU}}$$

where $\hat{\mathbf{P}} = P - \bar{\mathbf{P}}$ and $\bar{\mathbf{E}} = E + \bar{\mathbf{P}}$ and $\mathbf{E}_q = E + \mathbf{P}_q$ (since $h^{(0)} = \mathbf{x}\bar{\mathbf{E}} + \hat{\mathbf{P}}$).

$$\mathcal{M}(\mathbf{t}) = \sigma^*\Big(\underbrace{(x_{\text{query}}E + P_{\text{query}})\,QK^T\,(\mathbf{x}E + P)^T}_{\text{QK circuit}}/\sqrt{d}\Big) \cdot \underbrace{(\mathbf{x}E + P)\,VOU}_{\text{OV circuit}} + \underbrace{(x_{\text{query}}E + P_{\text{query}})\,U}_{\text{direct path}} \tag{2}$$

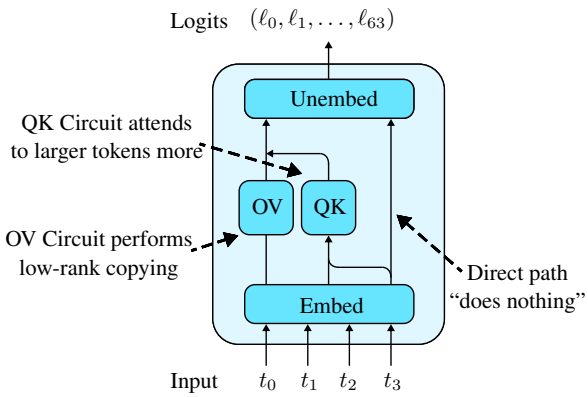


Figure 2: The models in our setting implement Max-of-K by attending exponentially more to larger tokens and copying the attended-to tokens (Section 3.1).

3.1. Mechanistic interpretation of learned models

Using standard empirical mechanistic interpretability techniques, we interpret one of our learned models (our "mainline" model) by independently examining the QK and OV circuits and the direct path.[4] We find that the model outputs the largest logit on the true max token $t_{\max}$ by attending more to larger tokens via the QK circuit and copying the tokens it attends to via the OV circuit. We then quantitatively confirm that these interpretations hold for all 151 models by reporting the mean plus minus standard deviation for various summary statistics. Plots for this section are available in Appendix B.2.

QK circuit By qualitatively examining the position-independent QK component EQKE, we find the amount of pre-softmax attention paid to a key token is approximately independent of the value of the query token t_{query}, and increases monotonically based on the size of the key token. We confirm this hypothesis by performing a singular-value decomposition (SVD) of the EQKE matrices (Appendix G.2.3), and find that it contains a single large rank-one component with singular value around 7800 ± 380, around 620 ± 130 times larger than the second largest component with singular value 13 ± 3. The left (query-side) singular vector is approximately constant in all dimensions, with value $0.1243 \pm 0.0003 \approx 1/8 = 1/\sqrt{d_{\text{vocab}}}$. The right (key-side) singular vector is monotonically increasing as we increase the size of the key token, with ($1/\sqrt{d}$-scaled) pre-softmax attention increasing by an average of 1.236 ± 0.056 when the key token increases by 1.[5]

In comparison, each $1/\sqrt{d}$-scaled entry of the position-dependent QK component EQKP has negligible size (average 0.31 ± 0.18), suggesting that EQKP is unimportant to the functioning of the model. We confirm this by zero ablating EQKP, which changes the models' accuracies from 0.9992 ± 0.0015 to 0.9993 ± 0.0011. Combined with our interpretation of EQKE, this implies that the attention pattern of the model depends only on the token values and not the ordering of the sequence.

OV circuit Then, by qualitatively examining the position-independent OV component EVOU, we see that it has large positive entries along the diagonal. In fact, the entry along the diagonal is the largest in the row for all rows corresponding to $t > 6.6 \pm 1.2$. Since each entry in the sequence is uniformly sampled and $d_{\text{vocab}} = 64$, this means that EVOU is a good approximation for the identity matrix for all but $\approx (7/64)^4 \approx 1.2 \times 10^{-2}\,\%$ of the sequences.

As with the position-dependent QK component, the position-dependent OV component PVOU also has negligible size and is unimportant to model performance. Taken together with the above results on EVOU, this suggests that the attention head copies the tokens it attends to.

Direct path As with the two position-dependent components, the entries in EU have small absolute magnitude 2.54 ± 0.20,[6] and contribute negligibly to model performance.

4. Proofs of model performance

In this section we describe intuitions for three categories of proof that are developed around different mechanistic interpretations and methods for using the interpretations. The strategies result in proofs of different complexities with varying bound tightness (Table 1). We provide detailed theorem statements, proofs, algorithms, and explanations of proof search in Appendices C, D, E, F and G.

Our theorem statements for Q will all be of the form

$$\forall \mathcal{M}', C_{\textit{specific strategy}}(\mathcal{M}') \leq \mathbb{E}_{\mathbf{t}\sim\mathcal{D}|_X} f(t_{\max}, \mathcal{M}'(\mathbf{t})).$$

We leave implicit the traces of running $C_{\textit{specific strategy}}$ on our

[4] All of our trained models behave similarly; see Appendix B.3.

[5] This implies that the ratio of attention paid to token t and $t-1$ is approximately $\exp(1.236) \approx 3.442$.

[6] For comparison, the average off-diagonal element of EVOU is 21.68 ± 0.83 below the corresponding diagonal element.

specific models to give the overall theorem. We report the computational complexity or estimated FLOPs of running $C_{\textit{specific strategy}}$ as approximations for our proof lengths.

4.1. The brute-force baseline

We start by considering the brute-force proof (Appendix D), which treats the model as a black box and evaluates it on all possible sequences.[7] However, this proof strategy has bad asymptotic complexity and is untenable for larger models and larger input distributions. So in subsequent sections, we use knowledge of the model drawn from the interpretation in Section 3.1 to derive more compact proofs.

4.2. A cubic proof

Next, we use the fact that the model is composed of the direct path and the QK and OV circuits (Section 3) to decrease the number of sequences that we need to consider, and the fact that only the position-independent components EQKE and EVOU contribute meaningfully to performance (Section 3.1) to pessimize over sequence ordering.

First, let a pure sequence ξ be a sequence with at most three distinct tokens: the max token $t_{\max}$, the query token $t_{\text{query}} \leq t_{\max}$, and optionally a third token $t' < t_{\max}$, and let Ξ^{pure} be the set of all pure sequences in X.[8] For a given input sequence $\mathbf{t}$, define the adjacent pure sequences $\text{Adj}(\mathbf{t})$ as the set of sequences that share the same max and query token, and only take on values in $\mathbf{t}$:

$$\text{Adj}(\mathbf{t}) = \left\{ \xi \in \Xi^{\text{pure}} \;\middle|\; \begin{array}{l} \max_i \xi_i = t_{\max},\ \xi_{\text{query}} = t_{\text{query}}, \\ \forall i < n_{\text{ctx}},\ \xi_i \in \mathbf{t} \end{array} \right\}$$

Using the convexity of softmax and the fact that the model contains three paths, we can show that one-layer attention-only transformers satisfies a variant of the following convexity property: for a given $\mathbf{t}$, if $\mathcal{M}(\xi)$ is correct for all $\xi \in \text{Adj}(\mathbf{t})$, then $\mathcal{M}(\mathbf{t})$ is correct. That is, for these transformers, we can bound the accuracy on all sequences by evaluating $\mathcal{M}$ on only the $O({d_{\text{vocab}}}^3(n_{\text{ctx}}-1)!)$ pure sequences. This allows us to bound the accuracy of our actual $\mathcal{M}$ on all ${d_{\text{vocab}}}^{n_{\text{ctx}}}$ sequences, while evaluating it on $O({d_{\text{vocab}}}^3(n_{\text{ctx}}-1)!)$ sequences.

We can reduce the number of sequences that we need to evaluate by pessimizing over the order of a sequence. For a given tuple of $(t_{\max}, t_{\text{query}}, t')$, there are $(n_{\text{ctx}}-1)!$ pure sequences, corresponding to the permutations of the tuple. Pessimizing over the order of sequences reduces the number of sequences to consider for each $(t_{\max}, t_{\text{query}}, t')$ tuple to the number of t' in the pure sequence, and the total number of sequences to $O({d_{\text{vocab}}}^3 n_{\text{ctx}})$. By precomputing the five component matrices EU, EQKE, EQKP, EVOU, PVOU and cleverly caching intermediate outputs, we can reduce the additional work of each sequence to the $O(n_{\text{ctx}})$ required to compute the softmax over n_{ctx} elements, resulting in asymptotic complexity $O({d_{\text{vocab}}}^3 {n_{\text{ctx}}}^2)$ (Theorem 23, additional details in Appendix E).

4.3. Sub-cubic proofs

We now consider proofs that are more compact than $O({d_{\text{vocab}}}^3)$. These require avoiding iteration over any set of size $O({d_{\text{vocab}}}^3)$ (e.g. the set of pure sequences) and performing operations that take $O(d_{\text{vocab}})$ time on each of $O({d_{\text{vocab}}}^2)$ combinations. Unfortunately, some methods of avoiding these operations can lead to vacuous bounds (i.e. accuracy lower bounds near 0%). In order to recover non-vacuous bounds, we introduce two tricks: the "mean+diff trick" to better approximate the sum of two components with unequal variance, and the "max row diff trick" to improve upon the low-rank approximations for EU and EQKE. We consider applying variants of these tricks at different locations in the naïve subcubic proof, leading to 100 distinct subcubic proof strategies. See Appendix G.2 for a formal description of these strategies.

4.3.1. Removing cubic-time computations

Reducing the number of cases by pessimizing over sufficiently small tokens Previously, we considered $\Theta({d_{\text{vocab}}}^3 n_{\text{ctx}})$ pure sequences ξ, with ξ parameterized by $(t_{\max}, t_{\text{query}}, t', c)$. Recall from our mechanistic interpretation in Section 3.1 that the pre-softmax attention paid from t_{query} to a key token t' is broadly invariant in t_{query} and increases roughly linearly with the size of t'. This allows us to pessimize over the OV circuit over all "sufficiently small" tokens.

More formally, suppose we are given some gap $g \in \mathbb{N}$. For each pure sequence ξ with max token $t_{\max}$, query token t_{query}, such that $t_{\text{query}} \leq t_{\max} - g$, and c copies of the third token type $t' \leq t_{\max} - g$, we pessimally ablate the OV circuit over the set $\Xi^{\text{pure}}(t_{\max}, t_{\text{query}}, c; g)$ of pure sequences ξ' with the same max and query tokens and c copies of the third token type t'. If the model gets all sequences in $\Xi^{\text{pure}}(t_{\max}, t_{\text{query}}, c; g)$ correct, then we can conclude that it gets ξ correct, otherwise, we treat the model as having gotten ξ wrong. This means that it suffices to only consider the $O({d_{\text{vocab}}}^2 n_{\text{ctx}})$ pessimal pure sequences of each of the $O({d_{\text{vocab}}}^2 n_{\text{ctx}})$ sets of the form $\Xi^{\text{pure}}(t_{\max}, t_{\text{query}}, c; g)$.

Decoupling and pessimizing computations that require $O({d_{\text{vocab}}}^3)$ computations Many parts of our cubic certificate require iterating through $O({d_{\text{vocab}}}^2)$ cases parameterized by $(t_{\max}, t_{\text{query}})$ or $(t_{\max}, t')$. For example, as part of the pessimization procedure over pure sequences, for

[7] Appendix A.6 discusses how to compute the "brute-force" accuracy of a model on an infinite distribution.

[8] In Section 4.3, we will consider a smaller set of "pure sequences".

Table 1: We report the proof complexity, normalized accuracy bound, and estimated flops required (Equation 2), as well as unexplained dimensionality (Section 5). We round the FLOP and unexplained dimension counts to the closest power of 2, and report the mean/standard deviation of the bound averaged across all 151 models. As we include more aspects of the mechanistic interpretation (reflected by a lower number of unexplained dimensions), we get more compact proofs (in terms of both asymptotic complexity and FLOPs), albeit with worse bounds. For space reasons, we use $k := n_{\text{ctx}}$, $d := d_{\text{model}}$, and $v := d_{\text{vocab}}$.

Description of Proof	Complexity Cost	Bound	Est. FLOPs	Unexplained Dimensions
Brute force	$\mathcal{O}(v^{k+1}kd)$	0.9992 ± 0.0015	2^{40}	2^{30}
Cubic	$\mathcal{O}(v^3k^2)$	0.9845 ± 0.0041	2^{25}	2^{14}
Sub-cubic	$\mathcal{O}(v^2 \cdot k^2 + v^2 \cdot d)$	0.832 ± 0.011	2^{21}	2^{13}
without mean+diff		0.758 ± 0.039	2^{21}	2^{13}
Low-rank QK	$\mathcal{O}(v^2k^2 + vd^2 + \text{(EU\&OV)}\ v^2d)$	0.806 ± 0.013	2^{21}	2^{12}
SVD only		0.643 ± 0.044	2^{22}	2^{12}
Low-rank EU	$\mathcal{O}(v^2k^2 + vd + \text{(QK\&OV)}\ v^2d)$	0.662 ± 0.061	2^{21}	2^{13}
SVD only		$(3.38 \pm 0.06) \times 10^{-6}$	2^{21}	2^{13}
Low-rank QK&EU	$\mathcal{O}(v^2k^2 + vd^2 + vd + \text{(OV)}\ v^2d)$	0.637 ± 0.060	2^{21}	2^{13}
SVD only		$(3.38 \pm 0.06) \times 10^{-6}$	2^{22}	2^{13}
Quadratic QK	$\mathcal{O}(v^2k^2 + vd + \text{(EU\&OV)}\ v^2d)$	0.407 ± 0.032	2^{21}	2^{12}
Quadratic QK&EU	$\mathcal{O}(v^2k^2 + vd + \text{(OV)}\ v^2d)$	0.303 ± 0.036	2^{21}	2^{13}

each of the d_{vocab} possible values of $t_{\max}$, we need to consider the relative effects on the d_{vocab}-sized logits of attending to each of the $O(d_{\text{vocab}})$ other tokens $t' < t_{\max}$, and for each $t_{\max}$ and t_{query}, we need to check that the contribution of the direct path on logits $x_{\text{query}}\text{EU}$ is not sufficiently large as to overwhelm the contribution from $x_{\max}\text{EVOU}$. We independently pessimize over each of these components over one of the d_{vocab}-sized axes: for example, instead of computing $x_{\max}\text{EVOU} + x_{\text{query}}\text{EU}$ for each $t_{\max}$, t_{query} pair, we first pessimally ablate the direct path along the query token (which takes $O({d_{\text{vocab}}}^2)$ time as it does not depend on the $t_{\max}$, and then consider the sum $x_{\max}\text{EVOU} + \max_{x'} x'\text{EU}$. Since this sum no longer depends on t_{query}, we only need to perform it $O(d_{\text{vocab}})$ times, for a total cost of $O({d_{\text{vocab}}}^2)$.

Low rank approximations to EQKE **and** EU Recall from Section 3.1 that EQKE is approximately rank 1, where the sole direction of variation is the size of the key token. By computing only the low rank approximation to EQKE, we can more cheaply compute the most significant component of the behavior in the QK circuit. To bound the remaining error, we can use the fact that after pulling off the first principal component from each of the four matrices we multiply, very little structure remains.

We can find the rank 1/2 approximations by performing SVD on EQKE. We can efficiently compute the SVD in $\mathcal{O}(d_{\text{vocab}}{d_{\text{model}}}^2)$ time by using the fact that EQKE can be written as the product of a $d_{\text{vocab}} \times d_{\text{model}}$ matrix and a $d_{\text{model}} \times d_{\text{vocab}}$ matrix. This allows us to avoid performing the $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}})$-cost matrix multiplications to explicitly compute EQKE.

Similarly, we can more efficiently check that the direct path EU contributes negligibly to the model outputs, by using SVD to decompose EU into a sum of rank 1 products (which we can evaluate exactly) and a high-rank error term that we can cheaply bound.

4.3.2. Additional subcubic proof strategies

Tighter bounds for sums of variables with unequal variance via the "mean+diff trick" Suppose we want to lower bound the minimum of the sum of two functions over three variables $h(x, y, z) = f(x, y) + g(y, z)$, while only iterating over two variables at a time. The naïve way is to minimize $f(x, y)$ and $g(x, y)$ independently:

$$\min_{x,y,z} h(x, y, z) \geq \min_{x,y} f(x, y) + \min_{y,z} g(y, z)$$

Here, the error comes from setting the ys in f and g to different values. But in cases where $g(y, z)$ varies significantly with y and only slightly with z, rewriting g as a sum of a component that is independent of z (only varying along y), and one that depends on z, yields a better lower bound:

$$\min_{x,y,z} h(x, y, z) \geq \min_{x,y} \left(f(x, y) + \mathbb{E}'_z g(y, z')\right) + \min_{y,z} (g(y, z) - \mathbb{E}'_z g(y, z'))$$

This estimate will have error at most ε, while the naïve estimator can have arbitrarily large error. We refer to this rewrite as the "mean+diff trick".[9] From the mechanistic interpretation in Section 3.1, we know that some of the components barely vary among one or more axes. So we can apply the mean+diff trick to get tighter lower bounds.

Avoiding matrix multiplications using the "max row-diff trick" Using properties of linear algebra, we derive a cheap approximation to the max row-diff for the product of matrices AB in terms of the product of the max row-diff of B and the absolute value of A, which we deem the "max row-diff" trick. We apply this trick to get a better cheap bound on the error terms of low-rank approximations, without having to multiply out the full matrices. See Appendix G.2.2 for details.

See Appendix F for more variants of these strategies.

5. Results

We run each of 151 transformers on the various proof strategies of different asymptotic complexity, and analyze these proofs to empirically examine the relationship between proof length, bound tightness, and degree of understanding. For each proof on each transformer, we approximate the length of the proof by estimating the number of FLOPs used, and plot this against the ratio of certified bound the true accuracy $b/\bar{s}$ (Equation 2) in Figure 3. There exists a clear trade-off between bound tightness and compactness of the proof – more compact proofs yield looser bounds, and tighter bounds are associated with more expensive proofs.

5.1. Compact proofs both require and provide mechanistic understanding

Quantifying mechanistic understanding using unexplained dimensionality We first quantify the amount of mechanistic understanding used in a proof by measuring its **unexplained dimensionality** – the number of free parameters required to fully describe model behavior, assuming the structural assumptions of the proof are correct. More detailed mechanistic interpretations will leave fewer free parameters that need to be filled in via empirical observation. (Details in Appendix A.2.) In Figure 5, we plot the two axes and find a suggestive correlation – that is, proofs based on less mechanistic understanding are longer.

More mechanistic understanding allows for more compact proofs In addition to the constructions in Section 4, the parts of proofs we were unable to compact seem to correspond to components that we do not mechanistically understand. For example, we could not cheaply bound the behavior of EVOU without multiplying out the matrices, and this seems in part because we do have a mechanistic understanding of how EVOU implements low-rank copying.

Compact proofs seem to provide understanding By examining compact proofs, we can extract understanding about the model. For example, the fact that replacing each row of EU with its average across rows has little effect on the bound implies that EU does not vary much based on t_{query}.

5.2. The trade-off between proof length and bound tightness is modulated by faithfulness of interpretation

Compact proofs are less faithful to model internals To derive more compact proofs, we use our mechanistic understanding to simplify the model computation in ways that diverge from the original model internals. For example, in some subcubic proofs (Section 4.3), we approximate EQKE with a rank-1 approximation corresponding to the "size direction". However, while other components are small, they're nonzero, and this approximation harms model internals.

Less faithful interpretations lead to worse bounds on performance To confirm that faithfulness of understanding affects the tightness of bound independent of proof length, we plot the normalized accuracy bound of subcubic proofs that perform a rank-1 approximation to EQKE, versus the ratio of the first two singular components. A larger ratio between the components implies that the rank-1 approximation is more faithful. In Figure 4, we see a positive correlation between the two axes: more faithful interpretations have tighter bounds for fixed proof length.

5.3. Compounding structureless noise poses a serious challenge to compacting proofs of global behavior

Pessimal error terms compound in the absence of known structure The rank-1 approximation of EQKE has small error. However, when making rank-1 approximations of each of the constituent matrices E, Q, K, pessimizing over the worst way to composing the individual small error terms leads to a bound on the error term of EQKE that is orders of magnitude larger than the actual error term. Because we don't understand how the matrices compose in a way that doesn't cause noise to compound (without just multiplying out the matrices), this approximation leads to a trivial bound on performance (Appendix G.2.5). We speculate that in many cases, there is no short human-interpretable description for why random noise or approximation errors do not compound across layers of neural networks (e.g., see the error correction results on *randomly initialized* neural networks from Vaintrob et al. (2023)), and thus that compounding structureless noise may be an issue in practice.

[9] In fact, this is the motivation behind the standard rewrites of QK and OV into position-independent and position-dependent components (Section 3).

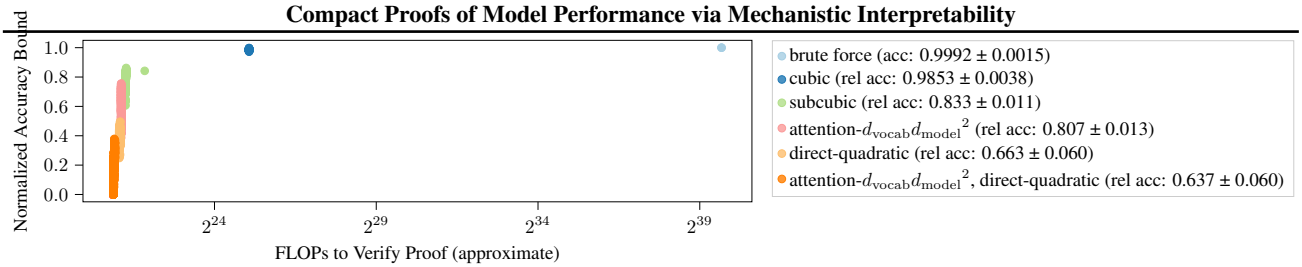


Figure 3: For each of the proofs in Section 4, we plot the number of FLOPs used to compute the certificate, as well as the normalized accuracy lower-bound ($b/\bar{s}$). The brute-force proof (Section 4.1) computes the exact performance uses orders of magnitude more compute than other approaches. The cubic proof (Section 4.3) uses a small amount of mechanistic understanding and less compute, while still retaining good accuracy lower bounds. Finally, subcubic proofs (Section 4.3) require the entirety of the mechanistic interpretation of the model to attain non-vacuous bounds; this understanding allows us to further reduces compute costs, but we still achieve worse bounds.

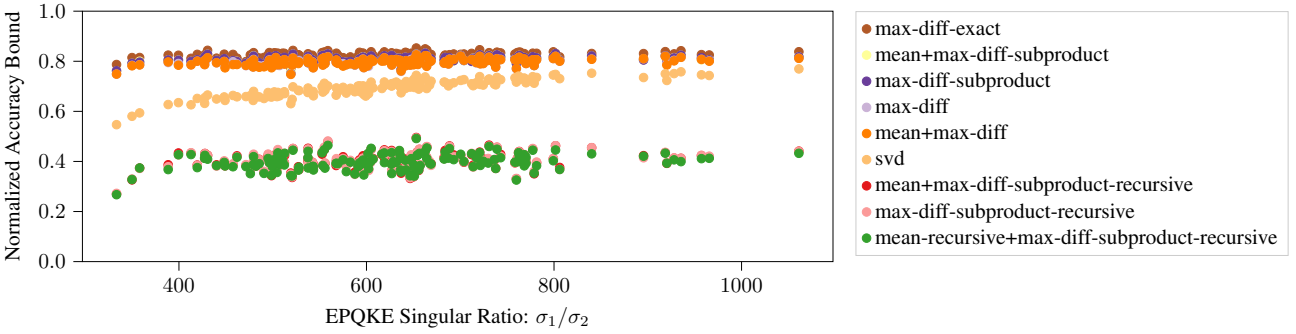


Figure 4: We plot the normalized accuracy bound versus the ratio of first and second singular values of EQKE, for various types of subcubic proofs that depend on a rank-1 approximation EQKE. For each class of proof, the closer EQKE is to rank-1, the tighter the accuracy bound. This suggests that more faithful interpretations lead to tighter bounds even holding proof length fixed.

6. Related Work

Generalization Bounds Prior work in the PAC-Bayes framework (Zhang et al., 2021; Nagarajan & Kolter, 2019; Dziugaite & Roy, 2017) proves generalization bounds over learning procedures, similar to the global performance bounds considered in this work. These proofs tend to provide statistical guarantees (Huang et al., 2018; 2023) about outputs of a known stochastic training procedure, while we seek to bound the performance of particular trained models.

Formally verifying neural networks Most prior work formally verifies neural networks either via model checking (Katz et al., 2017; Cheng et al., 2017) or by relaxing the problem setting and taking an automated theorem proving approach (Gehr et al., 2018; Singh et al., 2019; Gowal et al., 2018; Mirman et al., 2018; Raghunathan et al., 2018) to verify *local* robustness properties. These proof strategies tend to be derived by examining only the network architecture. We take an approach more akin to interactive theorem proving (Harrison et al., 2014) and verify *global* performance properties by reverse-engineering the neural network weights.

Mechanistic Interpretability Finally, mechanistic interpretability is the subfield of the broader field of understanding model internals (Räuker et al., 2022), which is too large to faithfully summarize. Our work takes most direct inspiration from efforts to deeply understand how either toy models (Nanda et al., 2023; Chughtai et al., 2023; Von Oswald et al., 2023; Akyürek et al., 2022) or small pretrained text transformers (Wang et al., 2022; Hanna et al., 2023) implement algorithmic tasks, generally by performing ablations and SVD. In contrast, we formally prove that a transformer implements an algorithm.

Nichani et al. (2024) proves that, in a significantly simplified 2-layer, 1-head attention-only transformer model and for the task of in-context bigram statistics, gradient descent will create induction heads (Olsson et al., 2022). Our results concern transformers with fixed weights. In concurrent work, Michaud et al. (2024) use techniques inspired by mechanistic interpretability to perform automated program synthesis on 2-dimensional RNNs, while our work works with significantly larger transformer models.

7. Conclusion and Future Work

Summary In this work, we used a Max-of-K setting to prototype the use of mechanistic interpretability to derive compact proofs of model behavior. Using varying amounts of understanding, we derived more efficient proof computa-

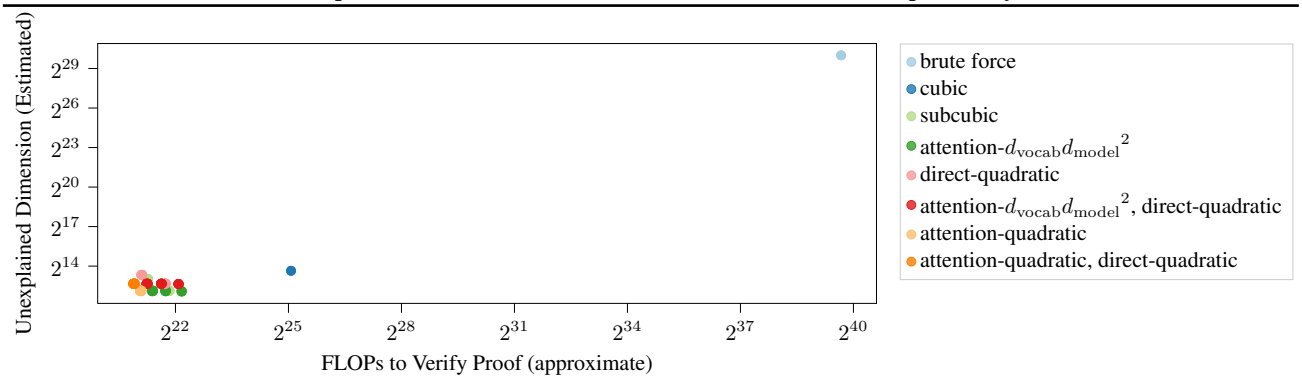


Figure 5: We plot, for each proof, the approximate number of flops required to evaluate the proof, versus the unexplained dimensionality (Section 5.1). More mechanistic understanding leaves fewer dimensions unexplained. We observe that more compact proofs seem to leave fewer unexplained dimensions, which is indicative of the relationship of mechanistic understanding and compact proofs.

tions lower bounding model accuracy. We found preliminary evidence that mechanistic understanding can compactify proofs. Moreover, we observed that the tightness of the lower bound offered by various proof strategies can be used to grade the faithfulness our mechanistic interpretation. Finally, we identified compounding structureless noise as a key obstacle to deriving compact proofs of model behavior.

Limitations and future work We study one-layer attention-only transformers on a toy algorithmic task. Future work should explore the viability of deriving proofs via interpretability using larger models featuring MLPs or layernorm on more complex domains. In addition, we were unable to significantly compact the part of the proof involving the OV circuit, which future work can explore. The proofs we explored in this work also did not lead to qualitatively novel insights; future work may be able to derive such insights with improved techniques. Finally, future work can address the problem of compounding structureless noise, perhaps by relaxing from worst-case pessimal ablations to typical-case heuristic guarantees (Christiano et al., 2022).

Acknowledgements

We are immensely grateful to Paul Christiano for providing the initial support for this project and for his invaluable research advice, encouragement, and feedback throughout its duration.

Additionally, we are thankful for clarifying discussions and feedback from Jacob Hilton, Matthew Coudron, Adrià Garriga-Alonso, Aryan Bhatt, Leo Gao, Jenny Nitishinskaya, Somsubhro Bagchi, Gabriel Wu, Erik Jenner, Ryan Greenblatt, Ronak Mehta, Louis Jaburi, and many others.

We are indebted to various organizations for their support:

- Alignment Research Center for funding this project and making it possible at all
- Mentorship for Alignment Research Students (MARS) program of the Cambridge AI Safety Hub (CAISH) for setting up the collaboration between a subset of authors, and providing funding for compute and in-person research sprints
- Constellation and FAR Labs for hosting a subset of the authors and providing an excellent research environment, including as part of the Visiting Fellows Program and Astra Fellowship

Author Contributions

Jason Gross led the project, including managing the team and conceptualizing the proofs approach. He ran the Max-of-4 experiments, devised the proof strategies, and wrote up the formal proofs. He worked on various case studies and developed general methodology for computing complexity and length bounds for proofs. He also developed the particular convex relaxations presented in the paper.

Rajashree Agrawal was invaluable in steering the direction of the project, including contributing to the preliminary experiment on Max-of-2 and developing the pessimal ablation approach. She worked on framing the results, and contributed text to the paper.

Thomas Kwa and **Euan Ong** extended the preliminary experiments to larger values of k and contributed substantially to the cubic proof. **Chun Hei Yip**, **Alex Gibson**, and **Soufiane Noubir** worked on case studies other than the Max-of-K task and informed discussion on proof complexity.

Lawrence Chan spearheaded the writing of the paper, including turning informal claims into formal theorem statements, creating figures, and writing the core text. He also developed the unexplained dimensionality metric for clarifying the takeaway of the paper.

References

Akbarpour, B., Abdel-Hamid, A., Tahar, S., and Harrison, J. Verifying a synthesized implementation of IEEE-754 floating-point exponential function using HOL. *The Computer Journal*, 53:465–488, May 2010. doi: 10.1093/comjnl/bxp023.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models, 2022.

Appel, A. and Kellison, A. VCFloat2: Floating-point error analysis in Coq. In *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2024, pp. 14–29, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704888. doi: 10.1145/3636501.3636953.

Boldo, S. and Melquiond, G. Flocq: A unified library for proving floating-point algorithms in Coq. In *2011 IEEE 20th Symposium on Computer Arithmetic*, pp. 243–252, July 2011. doi: 10.1109/ARITH.2011.40.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Chan, L., Garriga-Alonso, A., Goldwosky-Dill, N., Greenblatt, R., Nitishinskaya, J., Radhakrishnan, A., Shlegeris, B., and Thomas, N. Causal scrubbing, a method for rigorously testing interpretability hypotheses. *AI Alignment Forum*, 2022. URL https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing.

Cheng, C.-H., Nührenberg, G., and Ruess, H. Maximum resilience of artificial neural networks. In *Automated Technology for Verification and Analysis: 15th International Symposium, ATVA 2017, Pune, India, October 3–6, 2017, Proceedings 15*, pp. 251–268. Springer, 2017.

Christiano, P., Neyman, E., and Xu, M. Formalizing the presumption of independence. *arXiv preprint arXiv:2211.06738*, 2022.

Chughtai, B., Chan, L., and Nanda, N. A toy model of universality: Reverse engineering how networks learn group operations, 2023.

Dalrymple, D., Skalse, J., Bengio, Y., Russell, S., Tegmark, M., Seshia, S., Omohundro, S., Szegedy, C., Goldhaber, B., Ammann, N., et al. Towards guaranteed safe ai: A framework for ensuring robust and reliable ai systems. *arXiv preprint arXiv:2405.06624*, 2024.

Dziugaite, G. K. and Roy, D. M. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data, 2017.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Escardó, M. H. Synthetic topology of data types and classical spaces. *Electronic Notes in Theoretical Computer Science*, 87:21–156, November 2004.

Escardó, M. H. Infinite sets that admit fast exhaustive search. In *Proceedings of the 22nd Annual IEEE Symposium on Logic in Computer Science (LICS 2007)*, Wrocław, Poland, July 2007a.

Escardó, M. H. Seemingly impossible functional programs, 2007b. URL https://math.andrej.com/2007/09/28/seemingly-impossible-functional-programs/. Accessed: 2024-05-15.

Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., and Vechev, M. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, Los Alamitos, CA, USA, May 2018. IEEE Computer Society. doi: 10.1109/SP.2018.00058.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Hanna, M., Liu, O., and Variengien, A. How does gpt-2 compute greater-than. *Interpreting mathematical abilities in a pre-trained language model*, 2:11, 2023.

Harrison, J., Urban, J., and Wiedijk, F. History of interactive theorem proving. In *Handbook of the History of Logic*, volume 9, pp. 135–214. Elsevier, 2014.

Heimersheim, S. and Nanda, N. How to use and interpret activation patching. *arXiv preprint arXiv:2404.15255*, 2024.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Huang, X., Kroening, D., Ruan, W., Sharp, J., Sun, Y., Thamo, E., Wu, M., and Yi, X. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *arXiv preprint arXiv:1812.08342*, 2018.

Huang, X., Ruan, W., Huang, W., Jin, G., Dong, Y., Wu, C., Bensalem, S., Mu, R., Qi, Y., Zhao, X., Cai, K., Zhang, Y., Wu, S., Xu, P., Wu, D., Freitas, A., and Mustafa, M. A. A survey of safety and trustworthiness of large language models through the lens of verification and validation, 2023.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Katz, G., Barrett, C., Dill, D. L., Julian, K., and Kochenderfer, M. J. Reluplex: An efficient smt solver for verifying deep neural networks. In *Computer Aided Verification: 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I 30*, pp. 97–117. Springer, 2017.

Kellison, A. E., Appel, A. W., Tekriwal, M., and Bindel, D. LAProof: A library of formal proofs of accuracy and correctness for linear algebra programs. In *2023 IEEE 30th Symposium on Computer Arithmetic (ARITH)*, pp. 36–43, Los Alamitos, CA, USA, September 2023. IEEE Computer Society. doi: 10.1109/ARITH58626.2023.00021.

McKinney, W. Data Structures for Statistical Computing in Python. In van der Walt, S. and Millman, J. (eds.), *Proceedings of the 9th Python in Science Conference*, pp. 56–61, 2010. doi: 10.25080/Majora-92bf1922-00a.

Michaud, E. J., Liao, I., Lad, V., Liu, Z., Mudide, A., Loughridge, C., Guo, Z. C., Kheirkhah, T. R., Vukelić, M., and Tegmark, M. Opening the AI black box: program synthesis via mechanistic interpretability, 2024.

Mirman, M., Gehr, T., and Vechev, M. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pp. 3578–3586. PMLR, 2018.

Nagarajan, V. and Kolter, J. Z. Uniform convergence may be unable to explain generalization in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Nanda, N. and Bloom, J. Transformerlens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. *arXiv preprint*, 2023. doi: 10.48550/arXiv.2301.05217.

Nichani, E., Damian, A., and Lee, J. D. How transformers learn causal structure with gradient descent. *arXiv preprint*, 2024. doi: 10.48550/arXiv.2402.14735.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. PyTorch: An imperative style, high-performancedeep learning library. *Advances in neural information processing systems*, 32, 2019.

Plotly Technologies Inc. Collaborative data science, 2015. URL https://plot.ly.

Raghunathan, A., Steinhardt, J., and Liang, P. S. Semidefinite relaxations for certifying robustness to adversarial examples. *Advances in neural information processing systems*, 31, 2018.

Ramananandro, T., Mountcastle, P., Meister, B., and Lethin, R. A unified Coq framework for verifying C programs with floating-point computations. In *Proceedings of the 5th ACM SIGPLAN Conference on Certified Programs and Proofs*, CPP 2016, pp. 15–26, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341271. doi: 10.1145/2854065.2854066.

Räuker, T., Ho, A., Casper, S., and Hadfield-Menell, D. Toward transparent AI: A survey on interpreting the inner structures of deep neural networks. In *First IEEE Conference on Secure and Trustworthy Machine Learning*, 2022. doi: 10.48550/arxiv.2207.13243.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T., Eccles, T., Bruce, J., Razavi, A., Edwards, A., Heess, N., Chen, Y., Hadsell,

R., Vinyals, O., Bordbar, M., and de Freitas, N. A generalist agent, 2022. URL https://arxiv.org/abs/2205.06175.

Rogozhnikov, A. Einops: Clear and reliable tensor manipulations with Einstein-like notation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=oapKSVM2bcj.

Seshia, S. A., Sadigh, D., and Sastry, S. S. Toward verified artificial intelligence making ai more trustworthy with a formal methods-based approach to ai system verification and validation.

Simpson, A. K. Lazy functional algorithms for exact real functionals. In Brim, L., Gruska, J., and Zlatuška, J. (eds.), *Mathematical Foundations of Computer Science 1998*, pp. 456–464, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg. ISBN 978-3-540-68532-6.

Singh, G., Gehr, T., Püschel, M., and Vechev, M. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL), January 2019. doi: 10.1145/3290354.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Vaintrob, D., Hanni, K., and Mendel, J. Toward a mathematical framework for computation in superposition, 2023. URL https://www.lesswrong.com/posts/2roZtSr5TGmLjXMnT/toward-a-mathematical-framework-for-computation-in-superposition. Accessed: 2024-05-22.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint*, 2022. doi: 10.48550/arXiv.2211.00593.

Waskom, M. L. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Wong, E. and Kolter, Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International conference on machine learning*, pp. 5286–5295. PMLR, 2018.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

A. Subtleties of our approach

A.1. Convex relaxation

In this work, we construct convex relaxations to perform the pessimal ablations for our proofs.

In what sense are we using "convexity"? The intuition is that we are attempting to optimize a function f over its domain X by incrementally making local changes to the input sequence, such as replacing one token by another, or by changing the order of tokens. The reason that convex optimization problems are easy to solve is that all local extrema are global extrema. This is not the case for our optimization problem, so we find a relaxation of f and its domain such that all local extrema are in fact global extrema.

Furthermore, most convex optimizers perform optimization at runtime by repeatedly stepping towards extrema. In this work, we "optimize by hand", performing the optimization in the proof of our general theorems. The computation of the bound then only needs to instantiate the precomputed possible extrema with the actual values of the model's parameters to determine the the extrema actually are.

We now give a formal description of what we mean by "convex relaxation".

For a set of inputs X_i, we define a set of "relaxed inputs" X_i^{relaxed} with an injection $T_i : X_i \hookrightarrow \mathcal{P}(X_i^{\text{relaxed}})$ mapping input to the model to the set of corresponding relaxed inputs. On the relaxed input, we define a function $h_i : X_i^{\text{relaxed}} \to \mathbb{R}$ such that for all $\mathbf{t} \in X_i$ and all labels l for which $(l, \mathbf{t})$ is supported by (has non-zero probability in) $\mathcal{D}$, we can find $\mathbf{t}^{\text{relaxed}} \in T_i(\mathbf{t})$ with $f(l, \mathcal{M}(\mathbf{t})) \geq h_i(\mathbf{t}^{\text{relaxed}})$. We proceed by finding a small subset of "boundary" examples $B_i \subset X_i^{\text{relaxed}}$, proving that if $h_i(\mathbf{t}^{\text{relaxed}}) \geq b_i$ for all $\mathbf{t}^{\text{relaxed}} \in B_i$ then $h_i(\mathbf{t}^{\text{relaxed}}) \geq b_i$ for all $\mathbf{t}^{\text{relaxed}} \in X_i^{\text{relaxed}}$.

Then, the computational component C of the proof validates that that $h_i(\mathbf{t}^{\text{relaxed}}) \geq b_i$ for some b_i for all $\mathbf{t}^{\text{relaxed}} \in X_i^{\text{relaxed}}$. This allows us to conclude that $f(l, \mathcal{M}(\mathbf{t})) \geq b_i$ for all $\mathbf{t} \in X_i$.

A.2. Computing unexplained dimensionality

We claim in Figure 5 that we can use unexplained dimensionality as a metric for understanding. Here we describe how we compute the unexplained dimensionality of a proof strategy.

As in Figure 1, for any given proof, we can separate our treatment of transformer components into "black-box" (e.g., matrix multiplication) and "white-box" components (e.g., specifying that the QK circuit is approximately rank one; pessimizing over non-max tokens). Considering the performance score as a large white-box component which may reference black-boxes internally, we define the unexplained dimensionality of a single black-box computation as the log-cardinality of it function space (so, e.g, $2 \cdot 64$ for a function $\underline{64} \to \mathbb{R}^2$, whose cardinality is $(\mathbb{R}^2)^{\underline{64}}$, where $\underline{64}$ denotes the finite set on 64 elements). The unexplained dimensionality of the entire proof is the sum of the unexplained dimensions of all black-box components.

Intuitively speaking, unexplained dimensionality tries to capture the degrees of freedom that we have to check via brute enumeration over black-box computations. Proofs with less unexplained dimensionality contain more mechanistic understanding, and vice versa.

A.3. Computing approximate FLOPs

In Figure 3 and Table 1 on page 6 and on page 8, we display approximate floating point operations. We instrument our code to execute on phantom tensors that track their shape and accumulate an approximate count of floating point operations. We compute matrix additions and multiplications in the obvious way. We take the instruction count of SVD to be the cost of verifying that the output of SVD is a valid decomposition: that we have a pair of orthonormal bases which when multiplied out give the original basis.

A.4. IEEE 754 vs. $\mathbb{R}$

In Section 2 we defined C and Q and glossed over whether we were reasoning over reals or floats. Here we clarify this point that we've so far been sweeping under the rug.

Let $\mathbb{F}$ denote the set of the relevant flavor of IEEE 754 Floating Point numbers (generally 32-bit for our concrete models, but everything would hold just as well for 64-bit). Let $\mathbb{F}^*$ denote $\mathbb{F}$ restricted to finite numbers (that is, without NaNs and without $\pm\infty$).

We parameterize C, $\mathcal{M}$, and $\mathcal{D}$ over the real field[10] they operate on, so that, e.g., C_F : model weights $\to F$. Then we have Q establishing that for any model $\mathcal{M}'$, $C_{\mathbb{R}}(\mathcal{M}'_{\mathbb{R}}) \leq \mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}_{\mathbb{R}}} f_{\mathbb{R}}(l, \mathcal{M}'_{\mathbb{R}}(\mathbf{t}))$, and we have a trace demonstrating that $C_{\mathbb{F}}(\mathcal{M}_{\mathbb{F}}) = b$.

Let $i : \mathbb{F}^* \to \mathbb{R}$ be any injection that maps each floating point number to some real number that it is "closest to". Supposing that $b \in \mathbb{F}^*$ and thus $b \in \mathbb{R}$, we need two additional components of the proof. We need to find $\varepsilon, \varepsilon' \in \mathbb{R}^+$ prove that

$$|C_{\mathbb{R}}(\mathcal{M}_{\mathbb{R}}) - i\left(C_{\mathbb{F}}(\mathcal{M}_{\mathbb{F}})\right)| < \varepsilon \qquad \text{and} \qquad \left|\left(\mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}_{\mathbb{R}}} f_{\mathbb{R}}(l, \mathcal{M}_{\mathbb{R}}(\mathbf{t}))\right) - i\left(\mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}_{\mathbb{F}}} f_{\mathbb{F}}(l, \mathcal{M}_{\mathbb{F}}(\mathbf{t}))\right)\right| < \varepsilon'$$

Then we can chain these proofs to prove that

$$i\left(\mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}_{\mathbb{F}}} f_{\mathbb{F}}(l, \mathcal{M}_{\mathbb{F}}(\mathbf{t}))\right) \geq b - \varepsilon - \varepsilon'$$

Such ε-ball robustness proofs should be well within the scope of existing approaches to formal methods on neural nets, see, e.g., (Ramananandro et al., 2016; Appel & Kellison, 2024; Boldo & Melquiond, 2011; Kellison et al., 2023; Akbarpour et al., 2010; Wong & Kolter, 2018). We leave actually dealing with the gap between floating point numbers and real numbers to future work.

A.5. Non-uniform distributions

In Equation 1 in Section 2 we defined the expected model performance as the expectation of the distribution $\mathcal{D}$:

$$\bar{s} := \mathbb{E}_{(l,\mathbf{t})\sim\mathcal{D}}\left[f(l, \mathcal{M}(\mathbf{t}))\right] \geq b.$$

We then immediately specialized to the case where the marginalization $\mathcal{D}|_X$ of $\mathcal{D}$ over labels is uniform. As we'll see in Theorem 1 in Appendix D and Algorithm 3 in Appendix E, the bound computation is modularized between a function that bounds the performance $f(l, \mathcal{M}(\mathbf{t}))$ over a restricted collection of inputs, and a much simpler function that combines the bounds on individual cases into a bound on the expectation over the entire distribution. The per-input bound computation is CORRECTNESS in Algorithm 1 and RELAXED-CORRECTNESS-PESSIMIZING-OVER-POSITION in Algorithm 3; the expectation computation is BRUTE-FORCE in Algorithm 1 and CUBIC in Algorithm 3.

Since the expectation computation is modularized, it is straightforward to extend our approach to non-uniform distributions simply by adjusting the weighting of each region of inputs. However, if the distribution is too far off from the uniform training distribution, the bound we get may not be very good, as we may not be allocating adequate computation to the high-probability regions of the input space.

A.6. Infinite distributions

In the brute force proof in Section 4.1, we run the model on the entirety of $\mathcal{D}$. This operation is straightforward when X is finite. Perhaps surprisingly, we can do this even if X is infinite as long as the PDF $L \times X \to \mathbb{R}$ of $\mathcal{D}$ is computable and the natural computational topology of X is compact (Escardó, 2007b;a; 2004), because integration of computable functions on computable reals is computable (Simpson, 1998).

A.7. Using alternate loss functions

Building on the point from Appendix A.5, it is also relatively straightforward to extend our approach from bounding 0-1 loss (expected accuracy) to bounding log-loss. We will see in Figure 9 that the 0-1 loss and log-loss share a subterm $\Delta\ell_i$. Since we compute this subterm in all of our algorithms, we can easily extend our approach to log-loss by combining $\Delta\ell_i$ directly rather than merely checking that the value is negative as we currently do in RELAXED-CORRECTNESS-PESSIMIZING-OVER-POSITION in Algorithm 3. Although this is sufficient for the brute-force and cubic proofs, for the subcubic proof using Algorithm 5 in Appendix F, we would additionally have to compute a log-loss bound for the sequences where the largest non-max token is "too close" to the max token, which we currently neglect by considering the model to get them wrong in the worst case.

[10] Technically the floating point numbers are not a field. We gloss over this point here, since they define all of the field operations, even if those operations do not satisfy the field axioms.

A.8. Proving upper bounds

In this work, we focus on proving lower bounds on model performance. Most of our theorems, for example in Appendices E and F, prove two-sided bounds. Most of the other theorems can be straightforwardly adapted to proving upper bounds by swapping uses of $\min$ and $\max$. Therefore, we expect that proving upper bounds on model performance should be straightforward.

A.9. What proof system?

Length of proof depends on what proof system we use. We permit any proof system where proof-checking time is linear in the length of the proof.

B. Experimental details

B.1. Training details

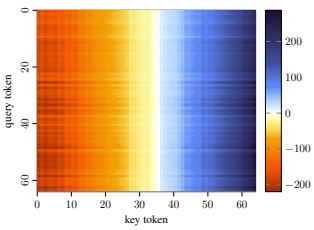
To train each model, we generate 384,000 random sequences of 4 integers picked uniformly at random, corresponding to less than 2.5% of the input distribution. We use AdamW with batch_size $= 128$, lr $= 0.001$, betas $= (0.9, 0.999)$, weight_decay left at the default 0.01. We train for 1 epoch (3000 steps). Over our 151 seeds, models trained with this procedure achieve $(99.92 \pm 0.15)\,\%$ train accuracy and a loss of 0.004 ± 0.008.[11] When qualitatively examining a single model (for example in Section 3.1 or Appendix H.1), we use the model with config seed 123, model seed[12] 613947648.

As our models as sufficiently small, we did not have to use any GPUs to accelerate training our inference. Each training run takes less than a single CPU-hour to complete. In total, the experiments in this paper took less than 1000 CPU-hours.

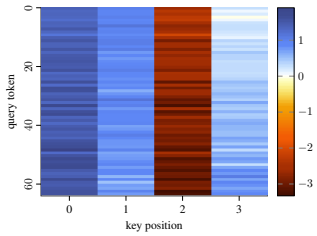
We use the following software packages in our work: Paszke et al. (2019); Plotly Technologies Inc. (2015); Nanda & Bloom (2022); Rogozhnikov (2022); Virtanen et al. (2020); McKinney (2010); Waskom (2021)

B.2. Additional details supporting our mechanistic interpretation of the model

We provide heatmaps of the matrices corresponding to the five components described/defined in Section 3, for the mainline model.



(a) $\mathrm{EQKE} = \mathbf{E}_q Q K^T \bar{\mathbf{E}}^T$

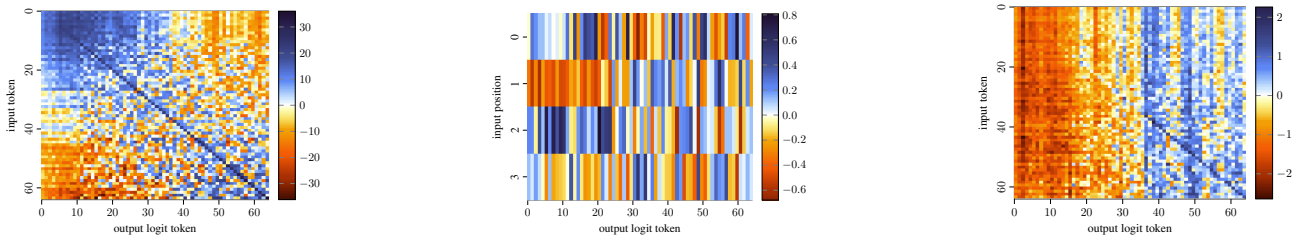


(b) $\mathrm{EQKP} = \mathbf{E}_q Q K^T \hat{\mathbf{P}}^T$

Figure 6: The QK circuit can be decomposed into the position-independent and position-dependent components EQKE and EQKP. It computes the pre-softmax attention score for the model. The positional contribution to the attention score, as shown in Figure (b), is minimal. In Figure (a), the gradient from left to right along the key axis indicates that the single attention head pays more attention to larger tokens. The uniformity along the query axis suggests that this behavior is largely independent of the query token. Further, the light and dark bands imply that some queries are better than others at focusing more on larger tokens.

[11] Numbers reported as mean across training runs $\pm$ std dev across training runs of mean accuracy and loss.

[12] The model seed is deterministically pseudorandomly derived from the seed 123.



(a) $\mathrm{EVOU} = \bar{\mathbf{E}}VOU$ (b) $\mathrm{PVOU} = \hat{\mathbf{P}}VOU$ (c) Direct Path $= \mathbf{E}_q U$

Figure 7: The OV circuit is a sum of EVOU and PVOU. In Figure (a) we see that EVOU "copies" — with the exception of input tokens ≤ 5 (6.6 ± 1.2 across all models) — by virtue of the fact that above 5, the diagonal is larger than all the other elements in the same row. We see that the range on Figure (b) is much smaller than Figure (a), indicating that positional contribution to the copying is minimal. In Figure (c) we see that direct path values matter a bit more than PVOU, being only $\approx 20\times$ smaller than the typical EVOU difference. They don't matter that much, though, being so small. Additionally, the vertical banding indicates that the primary effect of this is a largely-query-independent bias towards larger numbers, reflecting the fact that the input distribution is biased towards larger numbers being the maximum. The weak diagonal pattern indicates a slight bias towards upweighting the query token itself as a (possible) maximum token.

B.3. Distribution of model mechanisms

We provide some analysis of the distribution of the mechanisms of the models trained on the same configuration. At a glance, there is not that much variation across models.

The statistics of interest are: (1) σ_1/σ_2, the ratio of the first two singular values of EQKE, a measure of the extent to which the attention score computation is low-rank; (2) $\bar{s}$, the average score (accuracy) of the model across the entire input distribution; (3) $b_{\text{cubic}}/\bar{s}$, the percent-score-recovered accuracy bound achieved by the cubic proof from Section 4.2; (4) $b_{\text{subcubic}}/\bar{s}$, the percent-score-recovered accuracy bound achieved by the (per-model best)[13] subcubic proof from Section 4.3.

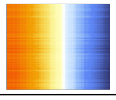
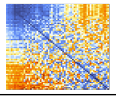
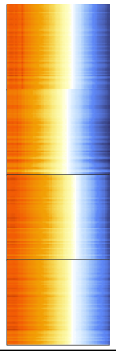
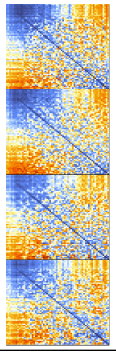
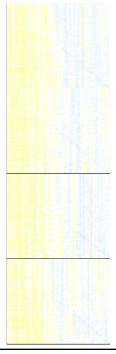
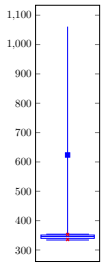
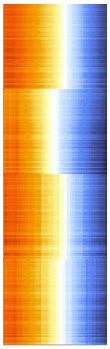
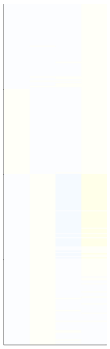
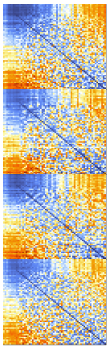
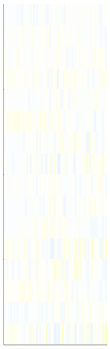
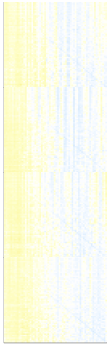
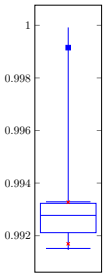
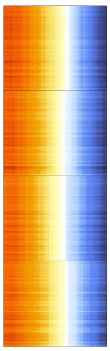
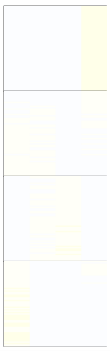
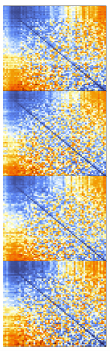
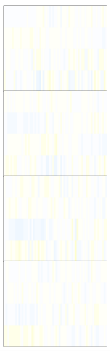
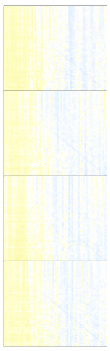
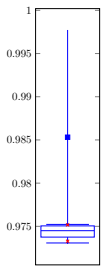
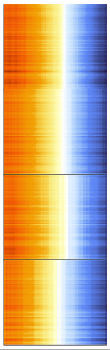
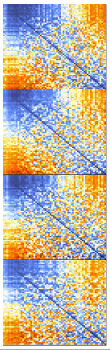
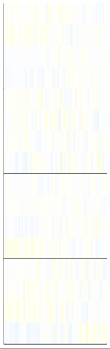
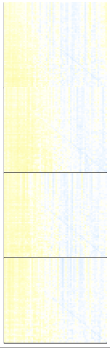
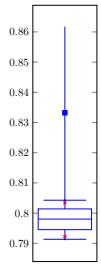
For each statistic of interest, Table 2 presents an eleven-number summary of the statistic. Plots, seeds, and statistic values are shown for models whose values are closest to each of the corresponding summary statistics.[14] Additionally, each group contains a boxplot of the summary:

- the minimum, maximum; the first and third quartiles; the median and mean; percentiles $2.15\,\%$, $97.85\,\%$, $8.87\,\%$, and $91.13\,\%$; these are displayed as:
- top and bottom of the vertical whisker lines; top and bottom of the box; horizontal line inside the box, and the square; horizontal whisker lines and whisker crosshatches.

[13] "Per-model best" here means that for each model seed, we select the variant of the subcubic proof with the highest bound.

[14] If a single model is the closest to two statistics, for example when the mean and median are very similar, the model is shown only once.

Table 2: Plots of various models. The statistics of interest are: (1) σ_1/σ_2, the ratio of the first two singular values of EQKE, a measure of the extent to which the attention score computation is low-rank; (2) $\bar{s}$, the average score (accuracy) of the model across the entire input distribution; (3) $b_{\text{cubic}}/\bar{s}$, the percent-score-recovered accuracy bound achieved by the cubic proof from Section 4.2; and (4) $b_{\text{subcubic}}/\bar{s}$, the percent-score-recovered accuracy bound achieved by the (per-model best) subcubic proof from Section 4.3. The y axes are: for EQKE and EQKP, the query token; for EVOU, the input token; for $\mathbf{E}_q U$, the input query token; for PVOU, the input position. The x axes are: for EQKE, the key token; for EQKP, the key position; for EVOU, PVOU, and $\mathbf{E}_q U$, the output logit token. All token axes range from 0 at the top (or left) to $d_{\text{vocab}} - 1 = 63$ at the bottom (or right). All position axes range from 0 at the top (or left) to $n_{\text{ctx}} - 1 = 3$ at the bottom (or right).

seed	statistic of interest distribution	value	EQKE $\mathbf{E}_q Q K^T \bar{\mathbf{E}}^T$	EQKP $\mathbf{E}_q Q K^T \hat{\mathbf{P}}^T$	EVOU $\bar{\mathbf{E}} VOU$	PVOU $\hat{\mathbf{P}} VOU$	$\mathbf{E}_q U$
			−400 −200 0 200 400		−40 −20 0 20 40		
123	(selected seed)						
24262	σ_1/σ_2	1061.1					
13654		626.5					
4810		350.0					
6204		333.4					
12457	$\bar{s}$	0.9999					
19451		0.9992					
15662		0.9932					
16197		0.9915					
1681	$b_{\text{cubic}}/\bar{s}$	0.998					
11598		0.985					
20754		0.975					
2306		0.973					
20415	$b_{\text{subcubic}}/\bar{s}$	0.86					
19004		0.83					
2306		0.80					
6155		0.79					

C. Mathematical definitions

We provide a detailed breakdown of the mathematical notation used in the appendix.

Let

$\underline{n}$	be the finite set on n elements; we write
$\mathbb{N}^{<n}$	$:= \{0, 1, \ldots, n-1\}$ when we care about the elements of $\underline{n}$
$\sigma(\mathbf{v})$	be the softmax function $e^{\mathbf{v}} / \sum_i e^{v_i}$
$\sigma^*(\mathbf{v})$	be the casually-masked softmax function, $\sigma^*(\mathbf{v})_i := e^{v_i} / \sum_{j \leq i} e^{v_j}$
d_{vocab}	be the size of the vocabulary
d	be the dimension of the attention head, in our case, equal to the hidden dimension of the model d_{model} (assumption: $d < d_{\text{vocab}}$)
n_{ctx}	be the context length, the number of tokens in the input sequence, equal to K in Max-of-K
P	be the $n_{\text{ctx}} \times d_{\text{model}}$ positional embedding
E	be the $d_{\text{vocab}} \times d_{\text{model}}$ token embed
Q, K, V, O	be the $d_{\text{model}} \times d_{\text{model}}$ query, key, value, and output matrices of the attention head
U	be the $d_{\text{model}} \times d_{\text{vocab}}$ unembed matrix
$\mathbf{t}$	be the input token sequence $[t_0, t_1, \ldots, t_{n_{\text{ctx}}-1}]$
$\mathbf{x}$	be the $n_{\text{ctx}} \times d_{\text{vocab}}$ one-hot-encoded input token sequence $[x_0, x_1, \ldots, x_{n_{\text{ctx}}-1}]$
x_{query}	$:= x_{-1} := x_{n_{\text{ctx}}-1}$ be the query token
t_{query}	$:= t_{-1} := t_{n_{\text{ctx}}-1}$ be the one-hot encoded query token
$t_{\max}$	be the true maximum token in the input sequence, $\max_i t_i$
$\mathcal{M}$	of type $X \to Y$ be the model[15]; sometimes we write
ℓ	for the logits of the model $\mathcal{M}(\mathbf{t})$
$\mathcal{D}$	be a probability distribution over input-label pairs $(\mathbf{t}, l) \in X \times L$
$\mathcal{D}\|_X$	be $\mathcal{D}$ marginalized to a distribution over $\mathbf{t} \in X$
f	of type $L \times Y \to \mathbb{R}$ be a scoring function for evaluating the performance of the model
P_{avg}	be the average position embeds across positions (of size d_{model}), $\frac{1}{n_{\text{ctx}}} \sum_i P_i$
$\bar{\mathbf{P}}$	be either $\mathbf{1}_{n_{\text{ctx}}} \otimes P_{\text{avg}}$ or $\mathbf{1}_{d_{\text{vocab}}} \otimes P_{\text{avg}}$ depending on context – that is the result of broadcasting P_{avg} back into the shape of P or E (that is, $n_{\text{ctx}} \times d_{\text{model}}$ or $d_{\text{vocab}} \times d_{\text{model}}$)
$\mathbf{P}_q$	be $\mathbf{1}_{d_{\text{vocab}}} \otimes P_{\text{query}}$, the broadcasting of P_{query}
$\hat{\mathbf{P}}$, $\bar{\mathbf{E}}$, $\mathbf{E}_q$	be $P - \bar{\mathbf{P}}$, $E + \bar{\mathbf{P}}$, and $E + \mathbf{P}_q$ respectively

For any vector-valued function $\mathbf{v}$ of length d_{vocab}, parameterized over the input sequence $\mathbf{t}$, let

$\Delta v_i := v_i - v_{t_{\max}}$ be the difference between the ith element of $\mathbf{v}$ and the element of $\mathbf{v}$ corresponding to the true maximum of the input sequence $t_{\max}$.

For our particular model, we will have

$$d_{\text{vocab}} := 64 \qquad d = d_{\text{model}} := 32 \qquad n_{\text{ctx}} := 4$$

$$X := (\mathbb{N}^{<d_{\text{vocab}}})^{n_{\text{ctx}}} \cong \underline{64}^4 \qquad L := \mathbb{N}^{<d_{\text{vocab}}} \cong \underline{64}$$

$\mathcal{D}|_X := U(0, 1, \ldots, d_{\text{vocab}}-1)^{n_{\text{ctx}}}$, the uniform distribution

Figure 8: Preliminary model definitions

$$\underbrace{\mathbb{E}_{((l,\mathbf{t}))\sim\mathcal{D}}\left[f(l,\mathcal{M}(\mathbf{t}))\right]}_{\bar{s}\text{ (average model score)}} \geq \underbrace{b}_{\text{lower bound}} \qquad \text{(theorem statement)} \tag{3}$$

We define the two typical performance functions corresponding to 0-1 accuracy and log-loss. Note the shared subterm $\Delta\ell_i$.

$$f^{\text{0-1}}(t_{\max},\ell) := \mathbb{1}[\operatorname*{argmax}_i \ell_i = t_{\max}] = \mathbb{1}[0 > \max_{i\neq t_{\max}} \underbrace{\ell_i - \ell_{t_{\max}}}_{\Delta\ell_i}] \tag{4}$$

$$f^{\text{log-loss}}(t_{\max},\ell) := (\sigma(\ell))_{t_{\max}} = 1/\textstyle\sum_i \exp(\underbrace{\ell_i - \ell_{t_{\max}}}_{\Delta\ell_i}) \tag{5}$$

We present the model definition in four different regroupings to define via underbrace labels various useful quantities:

$$\mathcal{M}(\mathbf{t}) = \ell(\mathbf{t}) = \sigma^*\Big(\underbrace{(x_{\text{query}}E + P_{\text{query}})\,QK^T\,(\mathbf{x}E+P)^T}_{\text{QK circuit}}/\sqrt{d}\Big)\cdot\underbrace{(\mathbf{x}E+P)\,VOU}_{\text{OV circuit}} + \underbrace{(x_{\text{query}}E + P_{\text{query}})\,U}_{\text{direct path}} \tag{6}$$

$$= \underbrace{\sigma^*\Big(x_{\text{query}}\Big(\underbrace{\mathbf{E}_qQK^T\bar{\mathbf{E}}^T}_{\text{EQKE}}\mathbf{x}^T + \underbrace{\mathbf{E}_qQK^T\hat{\mathbf{P}}^T}_{\text{EQKP}}\Big)/\sqrt{d}\Big)}_{\alpha^*(\mathbf{t})}\cdot\underbrace{\Big(\mathbf{x}\underbrace{\bar{\mathbf{E}}VOU}_{\text{EVOU}} + \underbrace{\hat{\mathbf{P}}VOU}_{\text{PVOU}}\Big)}_{\text{EPVOU}(\mathbf{t})} + x_{\text{query}}\underbrace{\mathbf{E}_qU}_{\text{EU}} \tag{7}$$

$$= \underbrace{\alpha^*(\mathbf{t})\cdot\mathbf{x}\bar{\mathbf{E}}VOU}_{\ell^{\text{EVOU}}(\mathbf{t})} + \underbrace{\alpha^*(\mathbf{t})\cdot\hat{\mathbf{P}}VOU}_{\ell^{\text{PVOU}}(\mathbf{t})} + \underbrace{x_{\text{query}}\mathbf{E}_qU}_{\ell^{\text{EU}}(x_{\text{query}})} \tag{8}$$

$$= \Big(\textstyle\sum_{i=0}^{n_{\text{ctx}}-1}\underbrace{\underbrace{(\alpha^*(\mathbf{t}))_i x_i\bar{\mathbf{E}}VOU}_{\ell^{\text{EVOU},i}(\mathbf{t})} + \underbrace{(\alpha^*(\mathbf{t}))_i\hat{\mathbf{P}}_iVOU}_{\ell^{\text{PVOU},i}(\mathbf{t})}}_{\ell^i(\mathbf{t})}\Big) + \underbrace{x_{\text{query}}\mathbf{E}_qU}_{\ell^{\text{EU}}(x_{\text{query}})} \tag{9}$$

Figure 9: Definitions of the model behavior

D. Brute-force proof

Theorem 1. *For* BRUTE-FORCE$(d_{\text{vocab}}, n_{\text{ctx}}, \mathcal{M})$ *as defined in Algorithm 1,*

$$\mathbb{E}_{\mathbf{t}\sim U(0,1,\ldots,d_{\text{vocab}}-1)^{n_{\text{ctx}}}}\left[\operatorname*{argmax}_i(\mathcal{M}(\mathbf{t}))_i = \max_i t_i\right] \geq \text{BRUTE-FORCE}(d_{\text{vocab}}, n_{\text{ctx}}, \mathcal{M})$$

Proof. In fact the two sides of the inequality are equal by definition. Hence the inequality follows by reflexivity of $\geq$. □

Algorithm 1 Counting Correct Sequences By Brute Force

```
function CORRECTNESS(M, input-sequence)
    return MODEL-BEHAVIOR(M, input-sequence) == MAX(input-sequence)
end function
function BRUTE-FORCE(d_vocab, n_ctx, M)
    return (1/(d_vocab^n_ctx)) SUM(CORRECTNESS(M, tokens) for tokens ∈ (RANGE(d_vocab))^n_ctx)
end function
```

E. Details of cubic proof

In this section, we prove formally the result used in Section 4.2, .

[15]Note that while in the main body, $\mathcal{M}(\mathbf{t})$ referred to the pre-softmax output logits, in the appendix we abuse notation and occasionally use it to refer to maximum token indicated by the logits where appropriate.

At its heart, the convexity of softmax[16] is an extension to a simple idea: a weighted average of scalar values is extremized by putting 100% of the weight on an extremal value.

Using this simple version of the theorem, however, gives a useless bound of 0% accuracy: if we pay no attention to the maximum of the sequence, of course we're going to get the wrong answer. Since in fact the space of possible weightings we may see in practice is much smaller (finite, in fact, with at most ${d_{\text{vocab}}}^{n_{\text{ctx}}}$ values), we may look for a more general version of this idea that gives us tighter bounds that still cover the space of possible weightings.

The weights are *not* linearly independently choosable (softmax is non-linear), so extremal values do not necessarily result from putting maximal attention on the worst token. It may be, when trying to find the worst case, that some positions are so dis-preferred that it makes more sense to choose a token that is "less bad" for those positions, if it draws enough attention away from the correct token. See Lemma 7 for details.

We thus spend this section characterizing a relaxation of the constraints on weights:

1. that contains all actually possible weightings,
2. that is extremized at weights that still correspond to some notion of "put the most weight on the extremal tokens", and
3. for which computing the extremal weightings is computationally efficient.

Before diving in, let's recall the proof that a weighted average of scalar values is extremized by putting 100% of the weight on extremal values:

Theorem 2 (Warmup: Extremizing weighted averages)**.** *Fix a set of values $v_i \in \mathbb{R}$. The weighted average is bounded by the extremal values: for any w_i such that $\sum_i w_i = 1$ and $0 \leq w_i \leq 1$,*

$$\min_i v_i \leq \sum_i w_i v_i \leq \max_i v_i$$

Proof. The proof is simple. We have

$$\sum_i w_i v_i - \min_i v_i = \sum_i w_i (v_i - \min_j v_j) \geq 0$$

and

$$\max_i v_i - \sum_i w_i v_i = \sum_i w_i (\max_j v_j - v_i) \geq 0$$

so the result follows. □

E.1. Proof strategy

The model computes the true maximum $t_{\max}$ when its outputs logits ℓ are such that $\Delta\ell_{t^*} := \ell_{t^*} - \ell_{t_{\max}} < 0$ for all $t^* \neq t_{\max}$.[17] As a result, it suffices to lower-bound the proportion of sequences where (an upper bound on) $\Delta\ell_{t^*}$ is negative for all $t^* \neq t_{\max}$. In particular, we will upper-bound the contribution from incorrect tokens t in positions i to the difference $\Delta\ell^i$ between incorrect (t^*) and correct ($t_{\max}$) output tokens $\Delta\ell^i_{t^*} = \ell^i_{t^*} - \ell^i_{t_{\max}}$.

We do this by arguing that the logit difference $\Delta\ell_{t^*}$ satisfies a certain notion of convexity over the space of a relaxation of sequences (Theorem 17), and constructing a set of $\Theta({d_{\text{vocab}}}^3 n_{\text{ctx}})$ "extremal" relaxed sequences where the position and token embedding components of attention are pessimized independently.

We start by first rewriting the contribution of each token through the attention head to the logit difference into the contributions involving PVOU and EVOU:

[16] See Appendix A.1 for the reason that we call this "convexity". Note that our use of "convexity" is purely descriptive in this section; all theorems are written out explicitly.

[17] We use the logit difference $\Delta\ell_{t^*}$ because: (a) it is shared in the computation of $f^{\text{0-1}}$ and $f^{\text{log-loss}}$; (b) it is a linear function of the various paths through the model, which can therefore be analyzed separately; (c) it leaves open both the options of pessimizing over output logit before or after combining contributions of various paths through the model.

$$\Delta\ell^{t}_{t^*}(\mathbf{t}) = \Delta\ell^{\mathrm{PVOU},i}{}_{t^*}(\mathbf{t}) + \Delta\ell^{\mathrm{EVOU},i}{}_{t^*}(\mathbf{t})$$

We then upper bound $\Delta\ell^{\mathrm{PVOU},i}{}_{t^*}(\mathbf{t})$ by noting that because the softmax attention is a weighted average of PVOU,

$$\begin{aligned}
\Delta\ell^{\mathrm{PVOU},i}{}_{t^*}(\mathbf{t}) &= \ell^{\mathrm{PVOU},i}(\mathbf{t})_{t^*} - \ell^{\mathrm{PVOU},i}(\mathbf{t})_{\max_j t_j} \\
&= \alpha_i^*(\mathbf{t})\mathrm{PVOU}_{i,t^*} - \alpha_i^*(\mathbf{t})\mathrm{PVOU}_{i,\max_j t_j} \\
&= \alpha_i^*(\mathbf{t})\left(\mathrm{PVOU}_{i,t^*} - \mathrm{PVOU}_{i,\max_j t_j}\right) \\
&\leq \alpha_i^*(\mathbf{t}) \max_i \left(\mathrm{PVOU}_{i,t^*} - \mathrm{PVOU}_{i,\max_j t_j}\right)
\end{aligned}$$

Since $\sum_i \alpha_i^*(\mathbf{t}) = 1$, we have

$$\sum_{i=0}^{n_{\mathrm{ctx}}-1} \Delta\ell^{\mathrm{PVOU},i}{}_{t^*}(\mathbf{t}) \leq \max_i \left(\mathrm{PVOU}_{i,t^*} - \mathrm{PVOU}_{i,\max_j t_j}\right)$$

We then construct a set Ξ^{pure} of "pure sequences" consisting of only three types of tokens in one of two orders, and show that for each input sequence $\mathbf{t}$ and readoff logit t^*, we bound the logit difference from the token embeddings $\Delta\ell^{\mathrm{EVOU},i}{}_{t^*}(\mathbf{t})$ using a small subset $\mathcal{X}$ of Ξ^{pure}:

$$\sum_{i=0}^{n_{\mathrm{ctx}}-1} \Delta\ell^{\mathrm{EVOU},i}{}_{t^*}(\mathbf{t}) \leq \max_{\xi\in\mathcal{X}} \sum_{i=0}^{n_{\mathrm{ctx}}-1} \Delta\ell^{\mathrm{EVOU},i}{}_{t^*}(\xi)$$

We construct a set X^{relaxed} of relaxed sequences, where each relaxed sequence $\mathbf{t}^{\mathrm{relaxed}}$ consists of a sequence and a position $(\mathbf{t}, i)$, where $\Delta\ell_{t^*}(\mathbf{t}, i)$ is evaluated by separately considering the positional contribution through attention (that is, the attention weighted PVOU) and the token contribution (that is, the attention-weighted EVOU) and direct contribution (the logit difference through the skip connection EU). Note that i indicates the position that we pay 100% of the attention to for the PVOU contribution.

We argue that $\Delta\ell_{t^*}(\mathbf{t}, i)$ satisfies a certain notion of convexity over mixtures of sequences, such that we can evaluate it only on a set of $\Theta({d_{\mathrm{vocab}}}^3 n_{\mathrm{ctx}})$ "extremal" sequences in a way that takes $O({d_{\mathrm{vocab}}}^3 n_{\mathrm{ctx}})$ total time to bound $\Delta\ell_{t^*}(\mathbf{t}, i)$ for *every* possible input sequence. We then use the extremal sequences that the model gets correct to lower bound the proportion of *all* sequences that the model will get correct. Specifically, we argue that Algorithm 3 provides a valid lower bound on the proportion of sequences the model gets correct.

E.2. Proof outline

We now proceed to the main results of this section.

Math fact: For each token t^*, the logit difference $\Delta\ell_{t^*}$ for any sequence $\mathbf{t}$ can be decomposed into the direct contribution from the embeds ℓ^{EU}, the attention-weighted position contribution (PVOU), and the attention-weighted token contribution (EVOU). Therefore, it suffices to upper bound each of the three components independently, since summing these upper bounds gives a valid upper bound on the logit difference.

We can compute the direct contribution ℓ^{EU} exactly by first computing $\mathrm{EU} = \mathbf{E}_q U$ and then, for each max, subtracting the logit of the max token from each row of the matrix. No theorems needed. For each max token, we can bound the position contribution by its maximum over positions (Theorem 17).

In order to upper bound the token contribution, we argue that any mixed sequence will be upper bounded by the maximum of the corresponding pure sequences (Theorem 18). We then argue that for pure sequences, it suffices to consider orderings where same tokens appear contiguously (Theorem 11).

E.3. Formal proof

For this subsection, all theorems are parameterized over the following quantities.

Definition 3 (Common theorem parameters)**.** Fix a token value function (à la a row difference in EVOU) $v : \mathbb{N}^{<d_{\text{vocab}}} \to \mathbb{R}$ and a token attention function (à la EQKE for a fixed query token) $a : \mathbb{N}^{<d_{\text{vocab}}} \to \mathbb{R}$. Fix a position value function (à la a row difference in PVOU) $w : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{R}$ and a position attention function (à la EQKP for a fixed query token) $b : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{R}$.

In practice, we'll take, for fixed query token t_{query}, fixed output token of interest t^*, and fixed maximum token $t_{\max}$,

$$v_t = \text{EVOU}_{t,t^*} - \text{EVOU}_{t,t_{\max}} \qquad a_t = \text{EQKE}_{t_{\text{query}},t}/\sqrt{d}$$
$$w_i = \text{PVOU}_{i,t^*} - \text{PVOU}_{i,t_{\max}} \qquad b_i = \text{EQKP}_{t_{\text{query}},i}/\sqrt{d}$$

Definition 4 (of a sequence via sorted tokens and a position permutation)**.** We can *define a sequence of tokens via sorted tokens and a position permutation* by specifying a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}^{<d_{\text{vocab}}}$ paired with a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$.

Definition 5 (sequence score)**.** Given a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}^{<d_{\text{vocab}}}$ and a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$ define the *sequence score* $s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma}$ as:

$$s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma} := \sum_{0 \leq i < n_{\text{ctx}}} v_{t_i} e^{a_{t_i} + b_{\sigma(i)}} \Bigg/ \sum_{0 \leq i < n_{\text{ctx}}} e^{a_{t_i} + b_{\sigma(i)}}$$

We will drop the token subscript, writing only s_σ, when the token values are unambiguous by context.

The sequence score here will be computing $\Delta\ell^{\text{EVOU}}{}_{t^*}$ for some fixed t^* and $t_{\max}$. The way we've set up our definitions, high scores predict t^* (and are thus bad), negative scores predict $t_{\max}$ (and are thus good), and more negative the scores, the stronger the prediction of $t_{\max}$.

Definition 6 (swap permutation)**.** Given a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$ of the n_{ctx} positions and two indices $0 \leq i, j < n_{\text{ctx}}$, define the *swap permutation* $\sigma_{i \leftrightarrow j}$ to be the permutation that is σ except swapping i and j:

$$\sigma_{i \leftrightarrow j}(k) = \begin{cases} \sigma(i) & \text{if } k = j \\ \sigma(j) & \text{if } k = i \\ \sigma(k) & \text{otherwise} \end{cases}$$

Define $\Delta_{\sigma, i \leftrightarrow j}$ to be the *difference in sequence scores when you swap i and j*:

$$\Delta_{\sigma, i \leftrightarrow j} := s_{\sigma_{i \leftrightarrow j}} - s_\sigma$$

Lemma 7 (Characterization of swapping tokens). Fix a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$. Fix $\sigma : \mathbb{N} \to \mathbb{N}$ be a permutation of the n_{ctx} positions. Fix indices $0 \leq i, j < n_{\text{ctx}}$.

Then there are two cases for $\text{sign}\left(\Delta_{\sigma, i \leftrightarrow j}\right)$:

1. If $a_{t_i} = a_{t_j}$ then $\text{sign}\left(\Delta_{\sigma, i \leftrightarrow j}\right) = -\text{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right) \text{sign}\left(v_{t_i} - v_{t_j}\right)$.
2. Otherwise, $\text{sign}\left(\Delta_{\sigma, i \leftrightarrow j}\right) = \text{sign}\left(a_{t_i} - a_{t_j}\right) \text{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right) \text{sign}\left(s_\sigma - \frac{v_{t_i} e^{a_{t_i}} - v_{t_j} e^{a_{t_j}}}{e^{a_{t_i}} - e^{a_{t_j}}}\right)$.

Intuitively, Lemma 7 says that, if the token contribution to attention is equal between tokens t_i and t_j, then the impact of swapping their positions $\sigma(i)$ and $\sigma(j)$ is entirely determined by how much attention is paid to the positions of i and j and the relative difference in their value. (Notably, by swapping these tokens, we don't affect the attention paid on other tokens, and so the effect of the change does not depend on the values of the other tokens.) Alternatively, if the attentions are not equal, then swapping the positions changes the allocation of attention to other tokens in the sequence, and so it may the case that this change in allocation in attention dominates the attention-weighted values of these two tokens.

Proof. First note that the theorem is trivial for $i = j$.

For the rest of the proof, we take $i \neq j$.

The proof proceeds just by algebraic manipulation with no deep insight. We first list the facts we use, the proceed to computing $\operatorname{sign}(\Delta_{\sigma,i\leftrightarrow j})$. We abbreviate $\sigma_{i\leftrightarrow j}$ as σ' for brevity.

$$\operatorname{sign}\left(e^{b_{\sigma(i)}} - e^{b_{\sigma(j)}}\right) = \operatorname{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right)$$

$$\begin{aligned}
\operatorname{sign}(\Delta_{\sigma,i\leftrightarrow j}) &= \operatorname{sign}(s_{\sigma'} - s_\sigma) \\
&= \operatorname{sign}\left(\frac{\sum_{0\le p<n_{\text{ctx}}} v_{t_p} e^{a_{t_p}+b_{\sigma'(p)}}}{\sum_{0\le p<n_{\text{ctx}}} e^{a_{t_p}+b_{\sigma'(p)}}} - s_\sigma\right)
\end{aligned}$$

Now multiply through by the denominator, which is positive

$$\begin{aligned}
&= \operatorname{sign}\left(\sum_{0\le p<n_{\text{ctx}}} v_{t_p} e^{a_{t_p}+b_{\sigma'(p)}} - s_\sigma \sum_{0\le p<n_{\text{ctx}}} e^{a_{t_p}+b_{\sigma'(p)}}\right) \\
&= \operatorname{sign}\Bigg(\sum_{0\le p<n_{\text{ctx}}} v_{t_p} e^{a_{t_p}+b_{\sigma(p)}} - v_{t_i} e^{a_{t_i}}\left(e^{b_{\sigma(i)}} - e^{b_{\sigma'(i)}}\right) - v_{t_j} e^{a_{t_j}}\left(e^{b_{\sigma(j)}} - e^{b_{\sigma'(j)}}\right) \\
&\qquad - s_\sigma \sum_{0\le p<n_{\text{ctx}}} e^{a_{t_p}+b_{\sigma(p)}} + s_\sigma e^{a_{t_i}}\left(e^{b_{\sigma(i)}} - e^{b_{\sigma'(i)}}\right) + s_\sigma e^{a_{t_j}}\left(e^{b_{\sigma(j)}} - e^{b_{\sigma'(j)}}\right)\Bigg) \\
&= \operatorname{sign}\Bigg(\cancel{\sum_{0\le p<n_{\text{ctx}}} v_{t_p} e^{a_{t_p}+b_{\sigma(p)}}} - v_{t_i} e^{a_{t_i}}\left(e^{b_{\sigma(i)}} - e^{b_{\sigma(j)}}\right) - v_{t_j} e^{a_{t_j}}\left(e^{b_{\sigma(j)}} - e^{b_{\sigma(i)}}\right) \\
&\qquad - \cancel{\sum_{0\le p<n_{\text{ctx}}} v_{t_p} e^{a_{t_p}+b_{\sigma(p)}}} + s_\sigma e^{a_{t_i}}\left(e^{b_{\sigma(i)}} - e^{b_{\sigma(j)}}\right) + s_\sigma e^{a_{t_j}}\left(e^{b_{\sigma(j)}} - e^{b_{\sigma(i)}}\right)\Bigg) \\
&= \operatorname{sign}\left(\left(v_{t_j} e^{a_{t_j}} - v_{t_i} e^{a_{t_i}}\right)\left(e^{b_{\sigma(i)}} - e^{b_{\sigma(j)}}\right) + s_\sigma\left(e^{a_{t_i}} - e^{a_{t_j}}\right)\left(e^{b_{\sigma(i)}} - e^{b_{\sigma(j)}}\right)\right) \\
&= \operatorname{sign}\left(e^{b_{\sigma(i)}} - e^{b_{\sigma(j)}}\right)\operatorname{sign}\left(\left(v_{t_j} e^{a_{t_j}} - v_{t_i} e^{a_{t_i}}\right) + s_\sigma\left(e^{a_{t_i}} - e^{a_{t_j}}\right)\right) \\
&= \operatorname{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right)\operatorname{sign}\left(s_\sigma\left(e^{a_{t_i}} - e^{a_{t_j}}\right) - \left(v_{t_i} e^{a_{t_i}} - v_{t_j} e^{a_{t_j}}\right)\right)
\end{aligned}$$

Divide through by non-zero values when possible

$$\begin{aligned}
&= \operatorname{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right) \\
&\quad \cdot \begin{cases} \operatorname{sign}\left(v_{t_i} - v_{t_j}\right) & \text{if } a_{t_i} = a_{t_j} \\ \operatorname{sign}\left(e^{a_{t_i}} - e^{a_{t_j}}\right)\operatorname{sign}\left(s_\sigma - \frac{v_{t_i} e^{a_{t_i}} - v_{t_j} e^{a_{t_j}}}{e^{a_{t_i}} - e^{a_{t_j}}}\right) & \text{otherwise} \end{cases} \\
&= \begin{cases} -\operatorname{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right)\operatorname{sign}\left(v_{t_i} - v_{t_j}\right) & \text{if } a_{t_i} = a_{t_j} \\ \operatorname{sign}\left(a_{t_i} - a_{t_j}\right)\operatorname{sign}\left(b_{\sigma(i)} - b_{\sigma(j)}\right)\operatorname{sign}\left(s_\sigma - \frac{v_{t_i} e^{a_{t_i}} - v_{t_j} e^{a_{t_j}}}{e^{a_{t_i}} - e^{a_{t_j}}}\right) & \text{otherwise} \end{cases}
\end{aligned}$$

□

Definition 8 (σ fixes F)**.** Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. ($F$ is the set of positions for which we are not pessimizing over the value of the token in that position.) Fix a non-decreasing sequence of tokens $t_0 \le \cdots \le t_{n_{\text{ctx}}-1} \in \mathbb{N}$.

Given a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{n_{\text{ctx}}}$, say that σ *fixes* F *(relative to* $t_0, \ldots, t_{n_{\text{ctx}}-1}$*)* if $t_i = t_F(\sigma(i))$ whenever $\sigma(i) \in F$.

Note that in this section, for the cubic proofs, we will in fact generally take $F = \{n_{\text{ctx}} - 1\}$, so that we are fixing the final query token, though in Theorems 18, 19 and 20 F will also contain all positions with the maximum token $t_{\max}$. In Appendix F, we will take $F = \emptyset$ or F to be the set of positions of the maximum token. However, none of these theorems are specific to F being subsingleton, and we prove them in generality.

Definition 9 (position-sorting permutation)**.** Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$.

Define the *position-sorting permutation fixing indices in* F $\sigma_s : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$ to be the permutation that sorts the indices not in F according to b: for $0 \leq i, j < n_{\text{ctx}}$ with $i, j \notin F$, $b_i \leq b_j$ whenever $\sigma_s(i) < \sigma_s(j)$; and $\sigma_s(i) = i$ for $i \in F$.

Definition 10 (contiguous on equal tokens)**.** Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. Fix a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$.

Say that the sequence represented by a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$ is *contiguous on equal tokens* if, for all $0 \leq i, j, k < n_{\text{ctx}}$ with $t_i = t_j \neq t_k$ and $i, j, k \notin \sigma^{-1}(F)$, it is never the case that $\sigma_s(\sigma(i)) < \sigma_s(\sigma(k)) < \sigma_s(\sigma(j))$.

Theorem 11 (Pessimization over sequence ordering is possible and results in contiguous sequences)**.** *Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. Fix a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$.*

Let $\sigma_{\min}, \sigma_{\max} : \mathbb{N} \to \mathbb{N}$ be permutations of the n_{ctx} positions, fixing positions in F, satisfying the following property: For all $\sigma : \mathbb{N} \to \mathbb{N}$ a permutation fixing F, we have

$$s_{\sigma_{\min}} \leq s_\sigma \leq s_{\sigma_{\max}} \tag{10}$$

(Such permutations are guaranteed to exist because the permutation group on n_{ctx} elements is finite.)

Then $\sigma_{\max}$ and $\sigma_{\min}$ may be taken to be contiguous on equal tokens. That is, there exist $\sigma_{\max}$ and $\sigma_{\min}$ satisfying the property of Equation 10 which additionally satisfy the definition of Theorem 10.

The basic idea is that we will assume that one of $\sigma_{\max}$ and $\sigma_{\min}$ cannot be contiguous on equal tokens and derive a contradiction. We will pick the extremal permutation that is closest to being contiguous, take a contiguity violation, and then show that either we can correct the contiguity violation without changing the score—thus violating the presumption that the permutation is *closest* to being contiguous—or we will find one swap of indices that decreases the score and another swap of indices that increases the score, thus violating the presumption of extremality.

In slightly more detail, but still informally, we will consider the sign of the difference between scores of our purported extremal permutation and a permutation that has swapped some indices. The theorem follows from showing that there exists a triple of indices i, j, k such that the sign of the score difference from swapping i and j is different from the sign of the score difference from swapping j and k.

First, a definition and some helpful facts about it.

Definition 12 (contiguous on equally-attended positions)**.** Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. Fix a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$.

Say that a permutation σ is *contiguous on equally-attended positions* if, for all $0 \leq i < n_{\text{ctx}}$ with $i \notin \sigma^{-1}(F)$, the sorting order according σ_s on the contiguous block of positions with contribution to the attention score equal to that of $\sigma(i)$, $\left\{\sigma(j) \,\middle|\, b_{\sigma(j)} = b_{\sigma(i)} \text{ and } \sigma(j) \notin F\right\}$, is the same as the sorting order according to the fraction of tokens equal to t_j with b-values greater than $b_{\sigma(i)}$, with ties broken by the value of t_j. Equationally, this second sorting order is defined by the score

$$\left(\left|\left\{k \,\middle|\, t_k = t_j \text{ and } b_{\sigma(k)} > b_{\sigma(i)} \text{ and } \sigma(k) \notin F\right\}\right| + \frac{t_j}{d_{\text{vocab}}}\right) \Big/ \left|\{k \,|\, t_k = t_j \text{ and } \sigma(k) \notin F\}\right| .$$

Most importantly, any permutation that is contiguous on equally-attended positions has the property that for any indices $0 \leq i, j, k < n_{\text{ctx}}$ with $i, j, k \notin \sigma^{-1}(F)$ and $t_i = t_j \neq t_k$ and $\sigma_s(\sigma(i)) < \sigma_s(\sigma(k)) < \sigma_s(\sigma(j))$, we will have the *strict* inequality $b_{\sigma(i)} < b_{\sigma(k)} < b_{\sigma(j)}$. Additionally, we may always sort equally-attended positions to make any permutation contiguous on equally-attended positions.

We will define an additional notion of contiguity-violations which we avoid up-front by arbitrarily swapping involved indices without changing the score s_σ.

Definition 13 (needlessly non-contiguous)**.** Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. Fix a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$.

Say that a permutation σ is *needlessly non-contiguous at i, j, k* (for $i, j, k \notin \sigma^{-1}(F)$) if $\Delta_{\sigma,i\leftrightarrow k} = 0$ or $\Delta_{\sigma,j\leftrightarrow k} = 0$, for $0 \leq i, j, k < n_{\text{ctx}}$ with $i, j, k \notin \sigma^{-1}(F)$ with $t_i = t_j \neq t_k$ and $\sigma_s(\sigma(i)) < \sigma_s(\sigma(k)) < \sigma_s(\sigma(j))$.

Say that a permutation σ is *needlessly non-contiguous* if it is needlessly non-contiguous at any $i, j, k \notin \sigma^{-1}(F)$.

Lemma 14. Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. Fix a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$.

Any needlessly non-contiguous sequence σ which fixes F can be made into a sequence σ' which still fixes F and is both simultaneously contiguous on equally-attended positions and not needlessly non-contiguous, and for which $s_\sigma = s_{\sigma'}$.

Proof. First, sort regions of equally-attended positions to make σ contiguous on equally-attended positions. If the resulting permutation is not needlessly non-contiguous, then we are done.

Otherwise, we have $\Delta_{\sigma,i\leftrightarrow k} = 0$ or $\Delta_{\sigma,j\leftrightarrow k} = 0$ for some i, j, k, for $0 \leq i, j, k < n_{\text{ctx}}$ with $i, j, n \notin \sigma^{-1}(F)$ and $t_i = t_j \neq t_k$ and $\sigma_s(\sigma(i)) < \sigma_s(\sigma(k)) < \sigma_s(\sigma(j))$. Since the sequence is contiguous on equally-attended positions, we have the strict inequality $b_{\sigma(i)} < b_{\sigma(k)} < b_{\sigma(j)}$.

By Lemma 7, we have two cases. Noting that $t_i = t_j$, we can write them as

1. $v_{t_k} = v_{t_i}$ and $a_{t_i} = a_{t_k}$
2. $a_{t_i} \neq a_{t_k}$ and $s_\sigma = \frac{v_{t_i} e^{a_{t_i}} - v_{t_k} e^{a_{t_k}}}{e^{a_{t_i}} - e^{a_{t_k}}}$

In the first case, we may fully freely interchange tokens equal to t_i with tokens equal to t_k without changing the score; in this case we may use the token value as a sorting tie-breaker and swap tokens until there are no more needlessly non-contiguous triples falling into case (1).

In the second case, since swapping tokens does not change s_σ, the property will continue to hold for these tokens after the swap. We may then swap tokens, again using token value as a tie-breaker, until there are no more needlessly non-contiguous triples falling into case (2). □

We can now finally make our argument for Theorem 11 more precise.

Proof of Theorem 11. Choose $\sigma_{\max}$ and $\sigma_{\min}$ to be contiguous on equally-attended positions and not needlessly non-contiguous, and suppose that we have $\sigma \in \{\sigma_{\max}, \sigma_{\min}\}$ such that for some $0 \leq i, j, k < n_{\text{ctx}}$ with $i, j, k \notin \sigma^{-1}(F)$ and $t_i = t_j \neq t_k$, we have $b_{\sigma(i)} < b_{\sigma(k)} < b_{\sigma(j)}$. We will derive a contradiction with the presumption that σ is extremal by showing that we can swap i and k to change the score in one direction and that we can swap j and k to change the score in the other direction.

Take σ_0' to be σ but swapping i and k, and take σ_1' to be σ but swapping j and k.

Now we will consider the cases for the sign of the score difference $\Delta_0 := s_{\sigma_0'} - s_\sigma$ and $\Delta_1 := s_{\sigma_1'} - s_\sigma$. By the presumption of not being needlessly non-contiguous, $\Delta_z \neq 0$ for $z \in \{0, 1\}$. If we can show that the sign of Δ_0 is distinct from the sign of Δ_1, then we will have a contradiction with extremality because we will have either $s_{\sigma_0'} < s_\sigma < s_{\sigma_1'}$ or $s_{\sigma_1'} < s_\sigma < s_{\sigma_0'}$. That is, we would be able to swap $i \leftrightarrow k$ and $j \leftrightarrow k$ to get a lower and higher score, making σ not extremal.

Noting that $t_i = t_j$,

$$\text{sign}\left(\Delta_0\right) = \text{sign}\left(b_{\sigma(i)} - b_{\sigma(k)}\right) \begin{cases} \text{sign}\left(v_{t_k} - v_{t_i}\right) & \text{if } a_{t_i} = a_{t_k} \\ \text{sign}\left(a_{t_i} - a_{t_k}\right) \text{sign}\left(s_\sigma - \frac{v_{t_i} e^{a_{t_i}} - v_{t_k} e^{a_{t_k}}}{e^{a_{t_i}} - e^{a_{t_k}}}\right) & \text{otherwise} \end{cases}$$

$$\text{sign}\left(\Delta_1\right) = \text{sign}\left(b_{\sigma(j)} - b_{\sigma(k)}\right) \begin{cases} \text{sign}\left(v_{t_k} - v_{t_i}\right) & \text{if } a_{t_i} = a_{t_k} \\ \text{sign}\left(a_{t_i} - a_{t_k}\right) \text{sign}\left(s_\sigma - \frac{v_{t_i} e^{a_{t_i}} - v_{t_k} e^{a_{t_k}}}{e^{a_{t_i}} - e^{a_{t_k}}}\right) & \text{otherwise} \end{cases}$$

Noting that the product is non-zero by presumption, that right multiplicand is equal for Δ_0 and Δ_1, and $\text{sign}\left(b_{\sigma(i)} - b_{\sigma(k)}\right) = -1$ and $\text{sign}\left(b_{\sigma(j)} - b_{\sigma(k)}\right) = 1$, we have our desired contradiction. □

Note that the proof of Theorem 11 does not go through if we include the position value function w in the score, because we may trade off the position value function against the token value function. We now show that we can *independently* pessimize over positional attention.

Definition 15 (full sequence score)**.** Given a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}^{<d_{\text{vocab}}}$ and a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$ define the *full sequence score* $s'_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma}$ as:

$$s'_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma} := \sum_{0\leq i<n_{\text{ctx}}} (v_{t_i} + w_{\sigma(i)}) e^{a_{t_i}+b_{\sigma(i)}} \Bigg/ \sum_{0\leq i<n_{\text{ctx}}} e^{a_{t_i}+b_{\sigma(i)}}$$

We will drop the token subscript, writing only s'_σ, when the token values are unambiguous by context.

The sequence score here will be computing $\Delta\ell^{\text{EPVOU}}_{t^*} := \Delta\ell^{\text{EVOU}}{}_{t^*} + \Delta\ell^{\text{PVOU}}{}_{t^*}$ for some fixed t^* and $t_{\max}$. As with Theorem 5, with the way we've set up our definitions, high scores predict t^* (and are thus bad), negative scores predict $t_{\max}$ (and are thus good), and more negative the scores, the stronger the prediction of $t_{\max}$.

Definition 16 (relaxed sequence score)**.** Given a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}^{<d_{\text{vocab}}}$ and a permutation $\sigma : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$ define the *relaxed sequence scores* $r_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma,\min}$ and $r_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma,\max}$ as:

$$r_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma,\min} := s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma} + \min_{0\leq i<n_{\text{ctx}}} w_i$$

$$r_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma,\max} := s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma} + \max_{0\leq i<n_{\text{ctx}}} w_i$$

We will drop the token subscript, writing only $r_{\sigma,\min}$ or $r_{\sigma,\max}$, when the token values are unambiguous by context.

Theorem 17 (Independent pessimization over positional contributions is possible)**.** *Fix non-decreasing sequences of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$ and $t'_0 \leq \cdots \leq t'_{n_{\text{ctx}}-1} \in \mathbb{N}$ and permutations $\sigma, \sigma' : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{N}^{<n_{\text{ctx}}}$. Let $r_{\sigma,\min}$ and $r_{\sigma,\max}$ denote $r_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma,\min}$ and $r_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma,\max}$; let s_σ denote $s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma}$; and let $s_{\sigma'}$ and $s'_{\sigma'}$ denote $s_{t'_0,\ldots,t'_{n_{\text{ctx}}-1},\sigma'}$ and $s'_{t'_0,\ldots,t'_{n_{\text{ctx}}-1},\sigma'}$.*

Then we have

$$\min_{0\leq i<n_{\text{ctx}}} w_i = r_{\sigma,\min} - s_\sigma \leq s'_{\sigma'} - s_{\sigma'} \leq r_{\sigma,\max} - s_\sigma = \max_{0\leq i<n_{\text{ctx}}} w_i.$$

That is, the difference between the relaxed sequence score and the sequence score of any given sequence always bounds the difference between the full sequence score and the sequence score for any (related or unrelated) sequence.

Proof. This proof follows straightforwardly from the softmax weighting being an affine weighting.

We have

$$\begin{aligned}
r_{\sigma,\min} - s_\sigma &= \min_{0\leq i<n_{\text{ctx}}} w_{\sigma(i)} = \min_{0\leq i<n_{\text{ctx}}} w_i \\
r_{\sigma,\max} - s_\sigma &= \max_{0\leq i<n_{\text{ctx}}} w_{\sigma(i)} = \max_{0\leq i<n_{\text{ctx}}} w_i \\
s'_{\sigma'} - s_{\sigma'} &= \sum_{0\leq i<n_{\text{ctx}}} w_{\sigma'(i)} e^{a_{t'_i}+b_{\sigma'(i)}} \Bigg/ \sum_{0\leq i<n_{\text{ctx}}} e^{a_{t'_i}+b_{\sigma'(i)}} \\
&= \sum_{0\leq i<n_{\text{ctx}}} w_{\sigma'(i)} \frac{e^{a_{t'_i}+b_{\sigma'(i)}}}{\sum_{0\leq j<n_{\text{ctx}}} e^{a_{t'_j}+b_{\sigma'(j)}}}
\end{aligned}$$

Since e^x is non-negative for all real x, we have

$$\min_{0\leq j<n_{\text{ctx}}} w_{\sigma'(j)} \frac{\sum_{0\leq i<n_{\text{ctx}}} e^{a_{t'_i}+b_{\sigma'(i)}}}{\sum_{0\leq j<n_{\text{ctx}}} e^{a_{t'_j}+b_{\sigma'(j)}}} \leq s'_{\sigma'} - s_{\sigma'} \leq \max_{0\leq j<n_{\text{ctx}}} w_{\sigma'(j)} \frac{\sum_{0\leq i<n_{\text{ctx}}} e^{a_{t'_i}+b_{\sigma'(i)}}}{\sum_{0\leq j<n_{\text{ctx}}} e^{a_{t'_j}+b_{\sigma'(j)}}}.$$

Thus we get as desired

$$\min_{0\leq i<n_{\text{ctx}}} w_i \leq s'_{\sigma'} - s_{\sigma'} \leq \max_{0\leq i<n_{\text{ctx}}} w_i.$$

□

Note that we could prove a more fine-grained theorem, that pessimizes over attention paid to positions only for sequences compatible with the chosen fixed tokens F and t_F, but since the positional contribution is so small we do not bother.

Theorem 18 (For a fixed ordering, softmax is convex over token counts and only pure sequences need be considered)**.** *Fix a set of fixed indices $F \subseteq \mathbb{N}^{<n_{\text{ctx}}}$ and an assignment of token values to each of the fixed positions $t_F : F \to \mathbb{N}^{<d_{\text{vocab}}}$. Fix a set $S \subseteq \mathbb{N}^{<d_{\text{vocab}}}$ of valid other tokens in the sequence. (In our uses of this theorem, S will be the largest subset of $\mathbb{N}^{\leq t_{\max}}$ for which we can guarantee that the model behaves correctly on all sequences compatible with F and $t_{\max}$ and with tokens otherwise drawn from S.)*

Define a comparison on non-negative integers less than d_{vocab}:

$$c := \sum_{i \in F} v_{t_F(i)} e^{a_{t_F(i)} + b_i} \qquad d := \sum_{i \in F} e^{a_{t_F(i)} + b_i} \qquad f := \sum_{\substack{0 \leq i < n_{\text{ctx}} \\ i \notin F}} e^{b_i}$$

$$\text{cmp}(x, y) := \text{sign}\left(d(e^{a_x} v_x - e^{a_y} v_y) - c(e^{a_x} - e^{a_y}) + f e^{a_x + a_y}\left(v_x e^{a_x + a_y} - v_y e^{a_x + a_y}\right)\right)$$

Let $t_{\text{cmp}\min}$ and $t_{\text{cmp}\max}$ be the minimum and maximum elements of S according to cmp.[18]

For a given choice of a non-decreasing sequence of tokens $t_0 \leq \cdots \leq t_{n_{\text{ctx}}-1} \in \mathbb{N}$ compatible with F and S and a given choice of permutation $\sigma : \mathbb{N} \to \mathbb{N}$ of the n_{ctx} positions fixing F ($t_i = t_F(\sigma(i))$ for $\sigma(i) \in F$; and $t_i \in S$ for $\sigma(i) \notin F$): let $s_{\sigma,\min}$ (and $s_{\sigma,\max}$) denote $s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma}$ when $t_i = t_{\text{cmp}\min}$ for all $\sigma(i) \notin F$ (or $t_{\text{cmp}\max}$, respectively).

Then for all such choices of sequence-permutation pairs,

$$s_{\sigma,\min} \leq s_{t_0,\ldots,t_{n_{\text{ctx}}-1},\sigma} \leq s_{\sigma,\max}.$$

This theorem follows by chaining two lemmas: that scores are extremized by considering pure sequences, and that the extremal pure sequences match the comparison function defined in the theorem statement.

Lemma 19 (Sequences scores are extremized on purer sequences). Fix all the same quantities as in Theorem 18.

For any indices $0 \leq i < j < n_{\text{ctx}}$, token values $x, y \in S$, the score for a sequence with $t_i = x \neq y = t_j$ is bounded on both sides by sequences with $t_i = t_j = x$ and $t_i = t_j = y$.

Proof. Let $s_{\alpha,\beta}$ be the sequence score with $t_i = \alpha$ and $t_j = \beta$, and define the score differences $\Delta_x := s_{x,x} - s_{x,y}$ and $\Delta_y := s_{y,y} - s_{x,y}$. It suffices to show that $\text{sign}(\Delta_x \Delta_y) \leq 0$. To show this, we must only compute the sign of Δ_α for $\alpha \in \{x, y\}$ and show that whenever both Δ_x and Δ_y are non-zero, they have opposite signs.

We proceed by computation after defining some convenience variables for brevity:

$$C := \sum_{\substack{0 \leq k < n_{\text{ctx}} \\ k \neq i,j}} v_{t_k} e^{a_{t_k} + b_{\sigma(k)}} \qquad D := \sum_{\substack{0 \leq k < n_{\text{ctx}} \\ k \neq i,j}} e^{a_{t_k} + b_{\sigma(k)}}$$

$$\tilde{\alpha} := \begin{cases} x & \text{if } \alpha = y \\ y & \text{if } \alpha = x \end{cases} \qquad i_\alpha := \begin{cases} i & \text{if } \alpha = x \\ j & \text{if } \alpha = y \end{cases} \qquad i_{\tilde{\alpha}} := \begin{cases} i & \text{if } \tilde{\alpha} = x \\ j & \text{if } \tilde{\alpha} = y \end{cases}$$

$$\begin{aligned} \text{sign}\left(\Delta_\alpha\right) &= \text{sign}\left(\frac{v_\alpha e^{a_\alpha + b_{\sigma(i)}} + v_\alpha e^{a_\alpha + b_{\sigma(j)}} + C}{e^{a_\alpha + b_{\sigma(i)}} + e^{a_\alpha + b_{\sigma(j)}} + D} - \frac{v_x e^{a_x + b_{\sigma(i)}} + v_y e^{a_y + b_{\sigma(j)}} + C}{e^{a_x + b_{\sigma(i)}} + e^{a_y + b_{\sigma(j)}} + D}\right) \\ &= \text{sign}\left(\frac{v_\alpha e^{a_\alpha + b_{\sigma(i_\alpha)}} + v_\alpha e^{a_\alpha + b_{\sigma(i_{\tilde{\alpha}})}} + C}{e^{a_\alpha + b_{\sigma(i_\alpha)}} + e^{a_\alpha + b_{\sigma(i_{\tilde{\alpha}})}} + D} - \frac{v_\alpha e^{a_\alpha + b_{\sigma(i_\alpha)}} + v_{\tilde{\alpha}} e^{a_{\tilde{\alpha}} + b_{\sigma(i_{\tilde{\alpha}})}} + C}{e^{a_\alpha + b_{\sigma(i_\alpha)}} + e^{a_{\tilde{\alpha}} + b_{\sigma(i_{\tilde{\alpha}})}} + D}\right) \end{aligned}$$

Multiply through by positive denominators and simplify

$$= \text{sign}\left(C\left(e^{a_{\tilde{\alpha}} + b_{\sigma(i_{\tilde{\alpha}})}} - e^{b_{\sigma(i_{\tilde{\alpha}})} + a_\alpha}\right) + D\left(v_\alpha e^{b_{\sigma(i_{\tilde{\alpha}})} + a_\alpha} - v_{\tilde{\alpha}} e^{a_{\tilde{\alpha}} + b_{\sigma(i_{\tilde{\alpha}})}}\right)\right.$$

[18] We will prove that cmp is transitive in the process of proving this theorem.

$$+ v_\alpha \left(e^{b_{\sigma(i_{\tilde{\alpha}})}} + e^{b_{\sigma(i_\alpha)}}\right) e^{a_{\tilde{\alpha}} + b_{\sigma(i_{\tilde{\alpha}})} + a_\alpha} - v_{\tilde{\alpha}} \left(e^{b_{\sigma(i_{\tilde{\alpha}})}} + e^{b_{\sigma(i_\alpha)}}\right) e^{a_{\tilde{\alpha}} + b_{\sigma(i_{\tilde{\alpha}})} + a_\alpha}\Big)$$

Pulling out $e^{b_{\sigma(i_{\tilde{\alpha}})}}$

$$= \operatorname{sign}\left(e^{a_{\tilde{\alpha}} + a_\alpha}\left(e^{b_{\sigma(i_{\tilde{\alpha}})}} + e^{b_{\sigma(i_\alpha)}}\right)(v_\alpha - v_{\tilde{\alpha}}) + C\left(e^{a_{\tilde{\alpha}}} - e^{a_\alpha}\right) + D\left(e^{a_\alpha} v_\alpha - e^{a_{\tilde{\alpha}}} v_{\tilde{\alpha}}\right)\right)$$

Note that swapping α and $\tilde{\alpha}$ negates the sign. Hence, we have $\operatorname{sign}(\Delta_x) = -\operatorname{sign}(\Delta_y)$ and hence $s_{x,x} \le s_{x,y} \le s_{y,y}$ or $s_{y,y} \le s_{x,y} \le s_{x,x}$ as desired. □

Lemma 20 (Pure sequences are sorted according to cmp in Theorem 18). Fix all the same quantities as in Theorem 18.

Fix tokens $x, y \in S$. Let $n := n_{\text{ctx}} - |F|$ be the number of non-fixed tokens. Fix sequences with n copies of x and y respectively: fix $t_{x,0} \le \cdots \le t_{x,n_{\text{ctx}}-1} \in \mathbb{N}$ and $t_{y,0} \le \cdots \le t_{y,n_{\text{ctx}}-1} \in \mathbb{N}$ compatible with F and S and given choices of permutations $\sigma_x, \sigma_y : \mathbb{N} \to \mathbb{N}$ of the n_{ctx} positions fixing F: $t_{x,i} = t_F(\sigma_x(i))$ for $\sigma_x(i) \in F$; $t_{y,i} = t_F(\sigma_y(i))$ for $\sigma_y(i) \in F$; $t_{x,i} = x$ for $\sigma_x(i) \notin F$; and $t_{y,i} = y$ for $\sigma_y(i) \notin F$.

Then

$$\operatorname{sign}((s_{\sigma_x, t_{x,0}, \ldots, t_{x,n_{\text{ctx}}-1}}) - (s_{\sigma_y, t_{y,0}, \ldots, t_{y,n_{\text{ctx}}-1}})) = \operatorname{cmp}(x, y)$$

Proof. The proof goes by straightforward computation.

$$\begin{aligned}
&\operatorname{sign}((s_{\sigma_x, t_{x,0}, \ldots, t_{x,n_{\text{ctx}}-1}}) - (s_{\sigma_y, t_{y,0}, \ldots, t_{y,n_{\text{ctx}}-1}})) \\
&= \operatorname{sign}\left(\frac{v_x e^{a_x} f + c}{e^{a_x} f + d} - \frac{v_y e^{a_y} f + c}{e^{a_y} f + d}\right)
\end{aligned}$$

Multiply through by non-negative denominators

$$\begin{aligned}
&= \operatorname{sign}\left((v_x e^{a_x} f + c)(e^{a_y} f + d) - (v_y e^{a_y} f + c)(e^{a_x} f + d)\right) \\
&= \operatorname{sign}\left(-cfe^{a_x} + cfe^{a_y} + dfv_x e^{a_x} - dfv_y e^{a_y} + f^2 v_x e^{a_x + a_y} - f^2 v_y e^{a_x + a_y}\right)
\end{aligned}$$

Use $f > 0$

$$\begin{aligned}
&= \operatorname{sign}\left(-ce^{a_x} + ce^{a_y} + dv_x e^{a_x} - dv_y e^{a_y} + fv_x e^{a_x + a_y} - fv_y e^{a_x + a_y}\right) \\
&= \operatorname{sign}\left(c\left(e^{a_y} - e^{a_x}\right) + d\left(v_x e^{a_x} - v_y e^{a_y}\right) + f\left(v_x e^{a_x + a_y} - v_y e^{a_x + a_y}\right)\right) \\
&= \operatorname{cmp}(x, y)
\end{aligned}$$

□

Corollary 21. Define the relation $\le_{\text{cmp}}$ by $x \le_{\text{cmp}} y$ if and only if $\operatorname{cmp}(x, y) \in \{-1, 0\}$. The relation $\le_{\text{cmp}}$ is always transitive.

Proof. Note that by Lemma 20, cmp is comparing two sequence scores. Since $\le$ is transitive over the reals, the relation $\le_{\text{cmp}}$ is also transitive. □

Finally, we combine the previous lemmas to complete our proof of Theorem 18:

Proof of Theorem 18. Extremal sequences with scores $s_{\sigma,\min}$ and $s_{\sigma,\max}$ are guaranteed to exist because there are only finitely many elements of S and therefore only finitely many sequences. By Lemma 19, the extremal sequences must be pure (have $t_i = t_j$ whenever $\sigma(i), \sigma(j) \notin F$). By Lemma 20, the extremal sequences must have tokens that are extremal according to cmp. □

Theorem 22.

$$\mathbb{E}_{\mathbf{t} \sim U(0,1,\ldots,d_{\text{vocab}}-1)^{n_{\text{ctx}}}}\left[\operatorname*{argmax}_i (\mathcal{M}(\mathbf{t}))_i = \max_i t_i\right] \ge \textsc{cubic}(d_{\text{vocab}}, n_{\text{ctx}}, \mathcal{M})$$

$$
\begin{aligned}
\mathrm{EQKE}(t_{-1}, t_i) &:= t_{-1}\mathbf{E}_q Q K^T \bar{\mathbf{E}}^T {t_i}^T / \sqrt{d} \\
\mathrm{EQKP}(t_{-1}, i) &:= t_{-1}\mathbf{E}_q Q K^T \hat{\mathbf{P}}_i^T / \sqrt{d} \\
\mathrm{EVOU}(t_i) &:= t_i \bar{\mathbf{E}} VOU \\
\mathrm{PVOU}(i) &:= \hat{\mathbf{P}}_i VOU \\
\ell^{\mathrm{EU}}(t_{-1}) &:= t_{-1}\mathbf{E}_q U \\
\Delta\ell^{\mathrm{EU}}{}_{t^*}(t_{-1}, \max_i t_i) &:= \ell^{\mathrm{EU}}(t_{-1})_{t^*} - \ell^{\mathrm{EU}}(t_{-1})_{\max_i t_i}
\end{aligned}
$$

Figure 10: Recapitulation of some relevant definitions from Figure 9, parameterized by the arguments they actually depend on.

Algorithm 2 Counting Correct Sequences in Cubic Time: Preliminaries

function CORRECTNESS($\mathcal{M}$, input-sequence)
  return MODEL-BEHAVIOR($\mathcal{M}$, input-sequence) == MAX(input-sequence)
end function
function MODEL-BEHAVIOR($\mathcal{M}$, input-sequence)
Require: input-sequence is a tensor of shape $(n_{\mathrm{ctx}},)$ with values in $\mathbb{N}^{<d_{\mathrm{vocab}}}$
  $t_{\max} \leftarrow$ MAX(input-sequence) ▷ $t_{\max} \leftarrow$ max-token
  $\mathbf{t} \leftarrow$ input-sequence
  skip-score$_{t^*} \leftarrow \Delta\ell^{\mathrm{EU}}{}_{t^*}(t_{n_{\mathrm{ctx}}-1}, t_{\max})$
  attn-weights-unscaled$_i \leftarrow \mathrm{EQKE}(t_{n_{\mathrm{ctx}}-1}, t_i) + \mathrm{EQKP}(t_{n_{\mathrm{ctx}}-1}, i)$
  attn-weights $\leftarrow$ SOFTMAX(attn-weights-unscaled$/\sqrt{d}$)
  $v_t \leftarrow \mathrm{EVOU}(t)$
  $w_i \leftarrow \mathrm{PVOU}(i)$
  $\Delta v_{t,t^*} \leftarrow v_{t,t^*} - v_{t,t_{\max}}$
  $\Delta w_{i,t^*} \leftarrow w_{i,t^*} - w_{i,t_{\max}}$
  return $\max_{t^* \neq t_{\max}}(\text{skip-score}_i + \sum_{i=0}^{n_{\mathrm{ctx}}-1}(\Delta v_{i,t^*} + \Delta w_{i,t^*}) \cdot \text{attn-weights}_i)$
end function
function CORRECTNESS-PESSIMIZING-OVER-POSITION-SLOW($\mathcal{M}$, input-sequence)
  $\mathbf{t} \leftarrow$ input-sequence
  return ALL(CORRECTNESS($\mathcal{M}$, perm + $[t_{-1}]$) **for all** perm $\in$ PERMUTATIONS($t_{0:-1}$))
end function

Algorithm 3 Counting Correct Sequences in Cubic Time. Lines are annotated with comments indicating the parameters for a cache to avoid duplicate computations.

function MODEL-BEHAVIOR-RELAXED($\mathcal{M}$, query-tok, max-tok, non-max-tok, n-copies-nonmax)
  $t_{\text{query}} \leftarrow$ query-tok, $t_{\max} \leftarrow$ max-tok, $t' \leftarrow$ non-max-tok, $c \leftarrow$ n-copies-nonmax
Require: $0 \leq t_{\text{query}} \leq t_{\max} < d_{\text{vocab}}$, $0 \leq t' \leq t_{\max} < d_{\text{vocab}}$, $0 \leq c < n_{\text{ctx}}$
Require: **if** n-copies-nonmax $= 0$ **then** non-max-tok $=$ max-tok
Require: **if** query-tok $\neq$ max-tok **then** n-copies-nonmax $< n_{\text{ctx}} - 1$
Ensure: **return** $\geq$ MODEL-BEHAVIOR($\mathcal{M}, \mathbf{t}$) **for all** $\mathbf{t}$ with specified t_{query}, c copies of t' in non-query positions, and the remainder of the tokens equal to $t_{\max}$
  skip-score$_{t^*} \leftarrow \Delta\ell^{\text{EU}}{}_{t^*}(t_{\text{query}}, t_{\max})$ ▷ Cache by $t_{\max}$, t_{query}, t^*
  $w_i \leftarrow \text{PVOU}(i)$ **for** $0 \leq i < n_{\text{ctx}}$ ▷ Cache by i
  $\Delta w_{\max,t^*} \leftarrow \max_{0 \leq i < n_{\text{ctx}}}(w_{i,t^*} - w_{i,t_{\max}})$ ▷ Cache by $t_{\max}$, t^*
  $v_t \leftarrow \text{EVOU}(t)$, $\Delta v_{t,t^*} \leftarrow v_{t,t^*} - v_{t,t_{\max}}$ **for** $t \in \{t_{\text{query}}, t_{\max}, t'\}$ ▷ Cache by $t_{\max}$, t, t^*
  $a_t \leftarrow \text{EQKE}(t_{\text{query}}, t)/\sqrt{d}$ **for** $t \in \{t_{\text{query}}, t_{\max}, t'\}$ ▷ Cache by t_{query}, t
  $b_{n_{\text{ctx}}-1} \leftarrow \text{EQKP}(t_{\text{query}}, n_{\text{ctx}} - 1)/\sqrt{d}$ ▷ Cache by t_{query}
  $b_{:-1} \leftarrow \text{SORT}(\text{EQKP}(t_{\text{query}}, :-1))/\sqrt{d}$ ▷ Cache by t_{query}, i
  attn-weights-unscaled$_{:,n_{\text{ctx}}-1} \leftarrow a_{t_{\text{query}}} + b_{n_{\text{ctx}}-1}$ ▷ Cache by t_{query}
  attn-weights-unscaled$_{0,i} \leftarrow a_{t_{\max}} + b_{i-c}$ **for** $c \leq i < n_{\text{ctx}} - 1$ ▷ Cache by t_{query}, $t_{\max}$, i, c
  attn-weights-unscaled$_{1,i} \leftarrow a_{t_{\max}} + b_i$ **for** $c \leq i < n_{\text{ctx}} - 1$ ▷ Cache by t_{query}, $t_{\max}$, i, c
  attn-weights-unscaled$_{0,i} \leftarrow a_{t'} + b_{c+i}$ **for** $0 \leq i < n_{\text{ctx}} - 1 - c$ ▷ Cache by t_{query}, t', i, c
  attn-weights-unscaled$_{1,i} \leftarrow a_{t'} + b_i$ **for** $0 \leq i < n_{\text{ctx}} - 1 - c$ ▷ Cache by t_{query}, t', i, c
  attn-weights$_0 \leftarrow$ SOFTMAX(attn-weights-unscaled$_0$) ▷ Cache by t_{query}, $t_{\max}$, t', i, c
  attn-weights$_1 \leftarrow$ SOFTMAX(attn-weights-unscaled$_1$) ▷ Cache by t_{query}, $t_{\max}$, t', i, c
  if $c = 0$ **then** ▷ In this case, attn-weights$_{0,i}$ = attn-weights$_{1,i}$, so we drop the first subscript
    return $\max_{t^* \neq t_{\max}}$(skip-score$_{t^*} + \Delta w_{\max,t^*} + \Delta v_{t_{-1},t^*}$attn-weights$_{-1} + \Delta v_{t_{\max},t^*} \sum_{i=0}^{n_{\text{ctx}}-2}$ attn-weights$_i$)
  else
    $\Delta v_{i,t^*} \leftarrow \Delta v_{t_{\max},t^*}$ **for** $c \leq i < n_{\text{ctx}} - 1$
    $\Delta v_{i,t^*} \leftarrow \Delta v_{t',t^*}$ **for** $0 \leq i < c$
    $\Delta v_{n_{\text{ctx}}-1,t^*} \leftarrow \Delta v_{t_{\text{query}},n_{\text{ctx}}-1}$
    return $\max_{t^* \neq t_{\max}}$ skip-score$_{t^*} + \max \begin{cases} \sum_{i=0}^{n_{\text{ctx}}-1} \max_{t^* \neq t_{\max}}(\Delta w_{\max,t^*} + \Delta v_{i,t^*}) \cdot \text{attn-weights}_{0,i} \\ \sum_{i=0}^{n_{\text{ctx}}-1} \max_{t^* \neq t_{\max}}(\Delta w_{\max,t^*} + \Delta v_{i,t^*}) \cdot \text{attn-weights}_{1,i} \end{cases}$
  end if
end function
function RELAXED-CORRECTNESS-PESSIMIZING-OVER-POSITION($\mathcal{M}$, t_{query}, $t_{\max}$, t', c)
  ▷ runs the model on a relaxed variant of input sequences compatible with the arguments
Ensure: **return** is False **if** CORRECTNESS-PESSIMIZING-OVER-POSITION-SLOW($\mathcal{M}$, $\mathbf{t}$) is False **for** *any* $\mathbf{t}$ with specified t_{query}, c copies of t' in non-query positions, and the remainder of the tokens equal to $t_{\max}$
  return MODEL-BEHAVIOR-RELAXED($\mathcal{M}, t_{\text{query}}, t_{\max}, t', c$) < 0
end function
function CUBIC(d_{vocab}, n_{ctx}, $\mathcal{M}$)
  count $\leftarrow 0$ ▷ # of correct sequences
  for $t_{\max} \in$ RANGE(d_{vocab}) **do** ▷ $t_{\max} \leftarrow$ max-token
    for $0 \leq t_{\text{query}} \leq t_{\max}$ **do** ▷ $t_{\text{query}} \leftarrow$ query-token
      $c_{\max} \leftarrow n_{\text{ctx}} - 1$ **if** $t_{\text{query}} = t_{\max}$ **else** $n_{\text{ctx}} - 2$ ▷ maximum copies of nonmax
      for $0 \leq c \leq c_{\max}$ **do** ▷ number of valid choices for the non-max token
        RCPOP($\vec{\chi}$) $\leftarrow$ RELAXED-CORRECTNESS-PESSIMIZING-OVER-POSITION($\mathcal{M}, \vec{\chi}$)
        if $c = 0$ **then**
          t-count $\leftarrow 1$ **if** RCPOP($t_{\text{query}}, t_{\max}, t_{\max}, 0$) **else** 0
        else
          t-count $\leftarrow \sum_{t'=0}^{t_{\max}-1} 1$ **if** RCPOP($t_{\text{query}}, t_{\max}, t', c$) **else** 0
        end if
        count $\leftarrow$ count $+ \binom{n_{\text{ctx}}-1}{c} \cdot$ (t-count)^c ▷ taking $0^0 = 0$ conventionally
      end for
    end for
  end for
  return count
end function

Proof. (sketch) Apply the previous theorems and lemmas to Algorithm 3. □

Theorem 23. *The running time of Algorithm 3, after using caching to avoid duplicate computations, is* $\mathcal{O}({d_{\text{vocab}}}^3 {n_{\text{ctx}}}^2)$.

Proof. The nested loops in CUBIC execute the innermost body $\mathcal{O}({d_{\text{vocab}}}^2 n_{\text{ctx}})$ times, and the summation on Line 42 costs $\mathcal{O}(n_{\text{ctx}})$ per iteration. What remains is to show that the call to RELAXED-CORRECTNESS-PESSIMIZING-OVER-POSITION$(\mathcal{M}, t_{\text{query}}, t_{\max}, t', c)$ costs $\mathcal{O}(n_{\text{ctx}})$ when $c \neq 0$ and at most $\mathcal{O}(d_{\text{vocab}} n_{\text{ctx}})$ when $c = 0$ and $t' = t_{\max}$.

The matrix multiplications in EQKE, EQKP, EVOU, PVOU, and ℓ^{EU} can be cached upfront, costing $\mathcal{O}(\max(d_{\text{vocab}}, d_{\text{model}}, n_{\text{ctx}})^2 d_{\text{model}}) \leq \mathcal{O}({d_{\text{vocab}}}^3)$ since we assume $d_{\text{vocab}} > d_{\text{model}}$ and $d_{\text{vocab}} > n_{\text{ctx}}$.

The sorting on Line 9 can also be cached upfront (per t_{query}), costing $\mathcal{O}(d_{\text{vocab}} n_{\text{ctx}} \log n_{\text{ctx}})$.

Note that each variable assignment in RELAXED-CORRECTNESS-PESSIMIZING-OVER-POSITION can be cached into a table parameterized over at most three variables which range over d_{vocab} and over at most two variables that range over n_{ctx}.

What remains is the **return** statements.

When $c = 0$, we have on Line 18: **return** $\max_{t^* \neq t_{\max}}(\text{skip-score}_{t^*} + \Delta w_{\max,t^*} + \Delta v_{t_{-1},t^*}\text{attn-weights}_{-1} + \Delta v_{t_{\max},t^*} \sum_{i=0}^{n_{\text{ctx}}-2} \text{attn-weights}_i)$. This is $\mathcal{O}(d_{\text{vocab}} n_{\text{ctx}})$ as desired.

When $c \neq 0$, we have on Line 23:

$$\textbf{return} \max_{t^* \neq t_{\max}} \text{skip-score}_{t^*} + \max \begin{cases} \sum_{i=0}^{n_{\text{ctx}}-1} \max_{t^* \neq t_{\max}}(\Delta w_{\max,t^*} + \Delta v_{i,t^*}) \cdot \text{attn-weights}_{0,i} \\ \sum_{i=0}^{n_{\text{ctx}}-1} \max_{t^* \neq t_{\max}}(\Delta w_{\max,t^*} + \Delta v_{i,t^*}) \cdot \text{attn-weights}_{1,i} \end{cases}$$

We can cache $\max_{t^* \neq t_{\max}} \text{skip-score}_{t^*}$ per $t_{\max}$ and t_{query}, costing $\mathcal{O}({d_{\text{vocab}}}^3 n_{\text{ctx}})$. We can cache $\max_{t^* \neq t_{\max}}(\Delta w_{\max,t^*} + \Delta v_{i,t^*})$ per $t_{\max}$ and t' costing $\mathcal{O}({d_{\text{vocab}}}^3)$, since each $\Delta v_{i,t^*}$ will be $\Delta v_{t,t^*}$ for some $t \in \{t_{\text{query}}, t_{\max}, t'\}$. Finally, we can compute the summation in cost $\mathcal{O}(n_{\text{ctx}})$ per loop iteration, as required. □

F. Quadratic counting for a sub-cubic proof

In this section we fill in the details lacking from Section 4.3.

In Appendix E we proved an intricate version of convexity of softmax where, modulo pessimizing in unrealistic ways over the attention paid to positions for the computation done on positional encodings, all extremal relaxed sequences correspond to actual sequences.

When we only get a budget of $\mathcal{O}({d_{\text{vocab}}}^2 n_{\text{ctx}})$ extremal relaxed cases to consider, though, we must pessimize more, which gives us a simpler version of the convexity theorem and proof. Notably, when we restrict our sequences to have only two tokens (the max token $t_{\max}$ and the non-max token t'), most of the theorems from Appendix E.3 get significantly simpler.

Additionally, we must pessimize separately over the token value (v) and token attention (b) computations in order to allow efficient computation (Theorem 31).

F.1. Proof of baseline sub-cubic result

For this subsection, all theorems are parameterized over the following quantities.

Definition 24 (Common theorem parameters)**.** Fix a total number of tokens n_{ctx}. Fix a token value function (à la a row-difference in EVOU) $v : \mathbb{N}^{<d_{\text{vocab}}} \to \mathbb{R}$ and a token attention function (à la EQKE for a fixed query token) $a : \mathbb{N}^{<d_{\text{vocab}}} \to \mathbb{R}$. Fix a position value function (à la a row-difference in PVOU) $w : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{R}$ and a position attention function (à la EQKP for a fixed query token) $b : \mathbb{N}^{<n_{\text{ctx}}} \to \mathbb{R}$.

In practice, as in Appendix E.3, we'll take, for fixed query token t_{query},

$$v_t = \text{EVOU}_{t,t^*} - \text{EVOU}_{t,t_{\max}} \qquad a_t = \text{EQKE}_{t_{\text{query}},t}/\sqrt{d}$$
$$w_i = \text{PVOU}_{i,t^*} - \text{PVOU}_{i,t_{\max}} \qquad b_i = \text{EQKP}_{t_{\text{query}},i}/\sqrt{d}$$

Note that unlike in Appendix E.3, we pessimize independently over the query token and the non-max token, so the "fixed" query token may not in fact appear in any key-side position in the relaxed sequence we consider.

Definition 25 (of a sequence via mapping from positions)**.** We can *define a sequence of tokens via mapping from positions* by specifying a subset of valid tokens $S \subseteq \mathbb{N}^{<d_{\text{vocab}}}$ paired with a function $T : \mathbb{N}^{<n_{\text{ctx}}} \to S$ specifying which token is in each position.

Definition 26 (sequence score)**.** Given a subset of valid tokens $S \subseteq \mathbb{N}^{<d_{\text{vocab}}}$ and a function $T : \mathbb{N}^{<n_{\text{ctx}}} \to S$ specifying which token is in each position, define the *sequence score*

$$s_T := \sum_{0 \le i < n_{\text{ctx}}} v_{T(i)} e^{a_{T(i)} + b_i} \Bigg/ \sum_{0 \le i < n_{\text{ctx}}} e^{a_{T(i)} + b_i}$$

Definition 27 (swapped mapping)**.** Given a subset of valid tokens $S \subseteq \mathbb{N}^{<d_{\text{vocab}}}$ and a function $T : \mathbb{N}^{<n_{\text{ctx}}} \to S$ specifying which token is in each position and two indices $0 \le i, j < n_{\text{ctx}}$, define the *swapped mapping* $T_{i \leftrightarrow j}$ be the function that is T except swapping i and j:

$$T_{i \leftrightarrow j}(k) = \begin{cases} T(i) & \text{if } k = j \\ T(j) & \text{if } k = i \\ T(k) & \text{otherwise} \end{cases}$$

Lemma 28 (Characterization of swapping tokens in a two-token sequence). Fix two tokens $t_0 < t_1 \in \mathbb{N}$ and a function $T : \mathbb{N}^{<n_{\text{ctx}}} \to \{t_0, t_1\}$ specifying which token is in each position.

Define $\Delta_{T, i \leftrightarrow j}$ to be the difference in sequence scores when you swap i and j:

$$\Delta_{T, i \leftrightarrow j} := s_{T_{i \leftrightarrow j}} - s_T$$

Then

$$\operatorname{sign}(\Delta_{T, i \leftrightarrow j}) = -\operatorname{sign}(b_i - b_j) \operatorname{sign}\left(v_{T(i)} - v_{T(j)}\right)$$

Proof. Lemma 7 gives us the result directly when $a_{T(i)} = a_{T(j)}$. Otherwise, we get

$$\operatorname{sign}(\Delta_{T, i \leftrightarrow j}) = \operatorname{sign}\left(a_{T(i)} - a_{T(j)}\right) \operatorname{sign}(b_i - b_j) \operatorname{sign}\left(s_T - \frac{v_{T(i)} e^{a_{T(i)}} - v_{T(j)} e^{a_{T(j)}}}{e^{a_{T(i)}} - e^{a_{T(j)}}}\right)$$

Hence all that remains is to show that

$$\operatorname{sign}\left(s_T \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - v_{T(i)} e^{a_{T(i)}} + v_{T(j)} e^{a_{T(j)}}\right) = -\operatorname{sign}\left(v_{T(i)} - v_{T(j)}\right)$$

Define $\bar{v} := \frac{1}{2}(v_{T(i)} + v_{T(j)})$ and define $\Delta v := \frac{1}{2}(v_{T(i)} - v_{T(j)})$ so that $v_{T(i)} = \bar{v} + \Delta v$ and $v_{T(j)} = \bar{v} - \Delta v$. Assume WLOG that $T(i) = 0$ and $T(j) = 1$ so that $v_{T(p)} = \bar{v} + (-1)^{T(p)} \Delta v$ for all p.

Then we have

$$\begin{aligned}
&\operatorname{sign}\left(s_T \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - v_{T(i)} e^{a_{T(i)}} + v_{T(j)} e^{a_{T(j)}}\right) \\
&= \operatorname{sign}\left(s_T \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - \bar{v}\left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - \Delta v e^{a_{T(i)}} - \Delta v e^{a_{T(j)}}\right) \\
&= \operatorname{sign}\left(\frac{\sum_{0 \le p < n_{\text{ctx}}} v_{T(p)} e^{a_{T(p)} + b_p}}{\sum_{0 \le p < n_{\text{ctx}}} e^{a_{T(p)} + b_p}} \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - \bar{v}\left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - \Delta v \left(e^{a_{T(i)}} + e^{a_{T(j)}}\right)\right) \\
&= \operatorname{sign}\left(\frac{\sum_{0 \le p < n_{\text{ctx}}} \left(\cancel{\bar{v}} + (-1)^{T(p)} \Delta v\right) e^{a_{T(p)} + b_p}}{\sum_{0 \le p < n_{\text{ctx}}} e^{a_{T(p)} + b_p}} \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - \cancel{\bar{v}\left(e^{a_{T(i)}} - e^{a_{T(j)}}\right)} - \Delta v \left(e^{a_{T(i)}} + e^{a_{T(j)}}\right)\right)
\end{aligned}$$

$$= \operatorname{sign}(\Delta v) \operatorname{sign}\left(\frac{e^{a_{T(i)}} \sum_{\substack{0 \le p < n_{\text{ctx}} \\ T(p)=T(i)}} e^{b_p} - e^{a_{T(j)}} \sum_{\substack{0 \le p < n_{\text{ctx}} \\ T(p)=T(j)}} e^{b_p}}{\sum_{0 \le p < n_{\text{ctx}}} e^{a_{T(p)}+b_p}} \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - e^{a_{T(i)}} - e^{a_{T(j)}} \right)$$

$$= \operatorname{sign}(v_{T(i)} - v_{T(j)}) \operatorname{sign}\left(\frac{e^{a_{T(i)}} \sum_{\substack{0 \le p < n_{\text{ctx}} \\ T(p)=T(i)}} e^{b_p} - e^{a_{T(j)}} \sum_{\substack{0 \le p < n_{\text{ctx}} \\ T(p)=T(j)}} e^{b_p}}{\sum_{0 \le p < n_{\text{ctx}}} e^{a_{T(p)}+b_p}} \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - e^{a_{T(i)}} - e^{a_{T(j)}} \right)$$

Define

$$P_i := \sum_{\substack{0 \le p < n_{\text{ctx}} \\ T(p)=T(i)}} e^{b_p} \qquad P_j := \sum_{\substack{0 \le p < n_{\text{ctx}} \\ T(p)=T(j)}} e^{b_p}$$

so that we get

$$\begin{aligned} &\operatorname{sign}\left(s_T \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - v_{T(i)} e^{a_{T(i)}} + v_{T(j)} e^{a_{T(j)}}\right) \\ &= \operatorname{sign}(v_{T(i)} - v_{T(j)}) \operatorname{sign}\left(\frac{e^{a_{T(i)}} P_i - e^{a_{T(j)}} P_j}{e^{a_{T(i)}} P_i + e^{a_{T(j)}} P_j} \left(e^{a_{T(i)}} - e^{a_{T(j)}}\right) - e^{a_{T(i)}} - e^{a_{T(j)}} \right) \end{aligned}$$

Multiply through by the positive denominator and expand out so that we get

$$\begin{aligned} &= \operatorname{sign}(v_{T(i)} - v_{T(j)}) \operatorname{sign}\left(-2e^{a_{T(i)}+a_{T(j)}} P_i - 2e^{a_{T(i)}+a_{T(j)}} P_j\right) \\ &= -\operatorname{sign}(v_{T(i)} - v_{T(j)}) \operatorname{sign}\left(e^{a_{T(i)}+a_{T(j)}} P_i + e^{a_{T(i)}+a_{T(j)}} P_j\right) \\ &= -\operatorname{sign}(v_{T(i)} - v_{T(j)}) \end{aligned}$$

□

Theorem 29 (Pessimization over sequence ordering for two-token sequences is simple)**.** *Let $\sigma_s : \mathbb{N} \to \mathbb{N}$ denote a permutation of the n_{ctx} positions that sorts them according to b: for $0 \le i, j < n_{\text{ctx}}$, $b_i \le b_j$ whenever $\sigma_s(i) < \sigma_s(j)$. Fix two tokens $t_0 < t_1 \in \mathbb{N}$.*

Let n_{t_0} be the number of $p \in [0, n_{\text{ctx}})$ with $T(p) = t_0$ and let n_1 be the number of $p \in [0, n_{\text{ctx}})$ with $T(p) = t_{t_1}$. Note that $n_{t_0} + n_{t_1} = n_{\text{ctx}}$.

Define $t_{\min} := \operatorname{argmin}_{t \in \{t_0, t_1\}} v_t$ and define $t_{\max} := \operatorname{argmax}_{t \in \{t_0, t_1\}} v_t$.

Define $T_{\min}, T_{\max} : \mathbb{N}^{<n_{\text{ctx}}} \to \{t_0, t_1\}$ to be the assignment of tokens to positions that pays the least (respectively, most) attention to $t_{\max}$:

$$T_{\min}(i) := \begin{cases} t_{\max} & \text{if } 0 \le \sigma_s(i) < n_{t_{\max}} \\ t_{\min} & \text{if } n_{t_{\max}} \le \sigma_s(i) < n_{\text{ctx}} \end{cases}$$

$$T_{\max}(i) := \begin{cases} t_{\min} & \text{if } 0 \le \sigma_s(i) < n_{t_{\min}} \\ t_{\max} & \text{if } n_{t_{\min}} \le \sigma_s(i) < n_{\text{ctx}} \end{cases}$$

Then we have that

$$s_{T_{\min}} \le s_T \le s_{T_{\max}}$$

Proof. The extremality of $s_{T_{\min}}$ and $s_{T_{\max}}$ follows straightforwardly from Theorem 11.

All that remains is $s_{T_{\min}} \leq s_{T_{\max}}$.

This follows from noting by Lemma 28 that swapping two tokens in $T_{\min}$ *increases* the sequence score, while the reverse is true of $s_{T_{\max}}$, thus showing that it must be $s_{T_{\min}}$ that is the minimum and $s_{T_{\max}}$ that is the maximum and not vice versa. □

Definition 30 (full sequence score)**.** Given a subset of valid tokens $S \subseteq \mathbb{N}^{<d_{\text{vocab}}}$ and a function $T : \mathbb{N}^{<n_{\text{ctx}}} \to S$ specifying which token is in each position define the *full sequence score* s'_T:

$$s'_T := \sum_{0 \leq i < n_{\text{ctx}}} (v_{T(i)} + w_i) e^{a_{T(i)} + b_i} \Bigg/ \sum_{0 \leq i < n_{\text{ctx}}} e^{a_{T(i)} + b_i}$$

Theorem 31 (Independent pessimization over positional contributions and token attention and token value is possible)**.** *Fix two tokens $t_0 < t_1 \in \mathbb{N}$. Let $T_{\min}, T_{\max} : \mathbb{N}^{<n_{\text{ctx}}} \to \{t_0, t_1\}$ and $t_{\max}$, $t_{\min}$ be as in Theorem 29. Fix a set S of valid tokens with $t_0, t_1 \in S$.*

Define relaxed versions $T'_{\max}, T'_{\min} : \mathbb{N}^{<n_{\text{ctx}}} \to S$ of $T_{\max}$ and $T_{\min}$:

$$T'_{\max}(i) := \begin{cases} T_{\max}(i) & \text{if } T_{\max}(i) = t_{\max} \\ \operatorname{argmin}_{\substack{j \in S \\ j \neq t_{\max}}} a_j & \text{otherwise} \end{cases}$$

$$T'_{\min}(i) := \begin{cases} T_{\min}(i) & \text{if } T_{\max}(i) = t_{\max} \\ \operatorname{argmax}_{\substack{j \in S \\ j \neq t_{\max}}} a_j & \text{otherwise} \end{cases}$$

That is, $T'_{\max}$ replaces $t_{\min}$ with whatever token in S draws the least attention away from $t_{\max}$, while $T'_{\min}$ replaces $t_{\min}$ with whichever token in S draws the most attention away from $t_{\max}$.

Define relaxed extremal sequence scores $r_{T_{\max}}$, $r_{T_{\min}}$:

$$r_{T_{\min}} := \min_{0 \leq i < n_{\text{ctx}}} w_i + \left(\sum_{0 \leq i < n_{\text{ctx}}} v_{T_{\min}(i)} e^{a_{T'_{\min}(i)} + b_i} \Bigg/ \sum_{0 \leq i < n_{\text{ctx}}} e^{a_{T'_{\min}(i)} + b_i} \right)$$

$$r_{T_{\max}} := \max_{0 \leq i < n_{\text{ctx}}} w_i + \left(\sum_{0 \leq i < n_{\text{ctx}}} v_{T_{\min}(i)} e^{a_{T'_{\max}(i)} + b_i} \Bigg/ \sum_{0 \leq i < n_{\text{ctx}}} e^{a_{T'_{\max}(i)} + b_i} \right)$$

Then $r_{T_{\min}} \leq s'_{T_{\min}}$ and $s'_{T_{\max}} \leq r_{T_{\max}}$.

Proof. (sketch) Essentially the same as the proof of Theorem 17. □

Note that in practice, we take S to be the set of all tokens less than $t_{\max} - g$ for some minimum gap g. This allows us to share computation across the various maximum tokens to reduce overall computational complexity.

Theorem 32. *For all G,*

$$\mathbb{E}_{\mathbf{t} \sim U(0,1,\ldots,d_{\text{vocab}}-1)^{n_{\text{ctx}}}} \left[\operatorname{argmax}_i (\mathcal{M}(\mathbf{t}))_i = \max_i t_i \right] \geq \textsc{subcubic}(d_{\text{vocab}}, n_{\text{ctx}}, \mathcal{M}, G)$$

Proof. (sketch) Apply preceding lemmas and theorems to Algorithm 5 □

Theorem 33. *The running time of Algorithm 5, after using caching to avoid duplicate computations, is $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}} + {d_{\text{vocab}}}^2 {n_{\text{ctx}}}^2)$.*

Proof. (sketch) Sum the complexities indicated along the right side of Algorithm 3. The ${d_{\text{vocab}}}^2 d_{\text{model}}$ term comes from the precomputing EVOU, EU, and EQKP. The ${d_{\text{vocab}}}^2 {n_{\text{ctx}}}^2$ term comes from the softmax over n_{ctx} tokens for $O({d_{\text{vocab}}}^2 n_{\text{ctx}})$ pessimized pure sequences. Confirming that none of the complexities on the right side exceeds $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}} + {d_{\text{vocab}}}^2 {n_{\text{ctx}}}^2)$ completes the proof.

□

Algorithm 4 Counting Correct Sequences in Subcubic Time, Preliminaries

function INPUT-SEQUENCE-COMPATIBLE-WITH(input-sequence, d_{vocab}, n_{ctx}, $t_{\max}$, t_{query}, c, g)
  $\mathbf{t} \leftarrow$ input-sequence
  return False **if** $\mathbf{t} \notin (\mathbb{N}^{<d_{\text{vocab}}})^{n_{\text{ctx}}}$ ▷ the sequence is not made of valid tokens
  return False **if** $t_{-1} \neq t_{\text{query}}$ ▷ wrong query token
  return False **if** $\max_i t_i \neq t_{\max}$ ▷ wrong max token
  return False **if** $|\{i \in \mathbb{N}^{<n_{\text{ctx}}} \mid t_i \neq t_{\max}\}| \neq c$ ▷ wrong count of non-max toks
  return ALL($t_i = t_{\max}$ **or** $t_{\max} - t_i \geq g$ **for** $0 \leq i < n_{\text{ctx}}$) ▷ check gap on non-max toks
end function
function CORRECTNESS-PESSIMIZING-OVER-GAP-SLOW($\mathcal{M}$, d_{vocab}, n_{ctx}, $t_{\max}$, t_{query}, c, g)
  return ALL(CORRECTNESS($\mathcal{M}$, $\mathbf{t}$) **for all** $\mathbf{t}$ s.t. INPUT-SEQUENCE-COMPATIBLE-WITH($\mathbf{t}$, d_{vocab}, n_{ctx}, $t_{\max}$, t_{query}, c, g))
end function
function SUBCUBIC(d_{vocab}, n_{ctx}, $\mathcal{M}$, G)
  count $\leftarrow 0$ ▷ # of correct sequences
  $G_{t_{\max},t_{\text{query}},c} \leftarrow \text{MIN}(t_{\max}, \text{MAX}(1, G_{t_{\max},t_{\text{query}},c}))$ ▷ Clip G to valid range
  $G^*_{t_{\max},c} \leftarrow \min_{t \leq t_{\max}} G_{t_{\max},t,c}$ ▷ Cache running minima
  for $t_{\max} \in \text{RANGE}(d_{\text{vocab}})$ **do** ▷ $t_{\max} \leftarrow$ max-token
    for $0 \leq t_{\text{query}} \leq t_{\max}$ **do** ▷ $t_{\text{query}} \leftarrow$ query-token
      $c_{\min} \leftarrow 0$ **if** $t_{\text{query}} = t_{\max}$ **else** 1 ▷ minimum copies of nonmax
      $c_{\max} \leftarrow \begin{cases} 0 & \textbf{if } t_{\max} = 0 \\ n_{\text{ctx}} - 1 & \textbf{if } t_{\max} = t_{\text{query}} \\ n_{\text{ctx}} - 2 & \textbf{else} \end{cases}$ ▷ maximum copies of nonmax
      for $c_{\min} \leq c \leq c_{\max}$ **do** ▷ number of valid choices for the non-max token
        $g \leftarrow G_{t_{\max},t_{\text{query}},c}$
        $g^* \leftarrow G^*_{t_{\max},c}$
        q-gap $\leftarrow t_{\max} - t_{\text{query}}$
        $\text{RCPOG}(\vec{\chi}) \leftarrow$ RELAXED-CORRECTNESS-PESSIMIZING-OVER-GAP($\mathcal{M}, \vec{\chi}$)
        if (q-gap $= 0$ **or** q-gap $\geq g$) **and** $\text{RCPOG}(d_{\text{vocab}}, n_{\text{ctx}}, t_{\max}, t_{\text{query}}, c, g, g^*)$ **then**
          $c' \leftarrow c$ **if** $t_{\text{query}} = t_{\max}$ **else** $c - 1$ ▷ # of non-max non-query tokens
          count $\leftarrow$ count $+ \binom{n_{\text{ctx}}-1}{c'}(t_{\max} - g)^{c'}$ ▷ taking $0^0 = 1$ conventionally
        end if
      end for
    end for
  end for
  return count
end function

Algorithm 5 Counting Correct Sequences in Subcubic Time

function MODEL-BEHAVIOR-RELAXED-OVER-GAP($\mathcal{M}$, t_{max}, t_{query}, c, g, g^*)
Ensure: CORRECTNESS-PESSIMIZING-OVER-GAP-SLOW is False $\implies$ result is False
Require: $0 \leq g^* \leq G \leq t_{\text{max}}$
Require: **if** $c = 0$ **then** $t_{\text{query}} = t_{\text{max}}$
    skip-score $\leftarrow \max_{t^*} \ell^{\text{EU}}(t_{\text{query}})_{t^*} - \min_{t^*} \ell^{\text{EU}}(t_{\text{query}})_{t^*}$ ▷ Cache by t_{query}
    $v_t \leftarrow \text{EVOU}(t)$
    $w_i \leftarrow \text{PVOU}(i)$
    $\Delta w_{\text{max},t^*} \leftarrow \max_i w_{i,t^*} - w_{p,t_{\text{max}}}$ ▷ Cache by t_{max}, t^*
    $\Delta w_{\text{max,max}} \leftarrow \max_{t^*} \Delta w_{\text{max},t^*}$ ▷ Cache by t_{max}
    $\Delta v_t \leftarrow \max_i v_{t,t^*} - \min_{t^*} v_{t,t^*}$ ▷ Cache by t
    $\Delta v_{\text{max}} \leftarrow \max_{0 \leq t \leq t_{\text{max}} - g^*} \Delta v_t$ ▷ Cache by t_{max}, c
    $\Delta v_{t^*}^{t_{\text{max}}} \leftarrow v_{t_{\text{max}},t^*} - v_{t_{\text{max}},t_{\text{max}}}$ ▷ Cache by t_{max}
    $\Delta v_{\text{max}}^{t_{\text{max}}} \leftarrow \max_{t^* \neq t_{\text{max}}} \Delta v_{t^*}^{t_{\text{max}}}$ ▷ Cache by t_{max}
    if $c = 0$ **then**
        $\ell_{t^*} \leftarrow \ell^{\text{EU}}(t_{\text{max}})_{t^*} + v_{t_{\text{max}},t^*} + \Delta w_{\text{max},t^*}$
        return $\max_{t^* \neq t_{\text{max}}} (\ell_{t^*} - \ell_{t_{\text{max}}})$
    end if
    $b_{:,n_{\text{ctx}}-1} \leftarrow \text{EQKP}(t_{\text{query}}, n_{\text{ctx}} - 1)/\sqrt{d}$ ▷ Cache by t_{query}
    $b_{0,:-1} \leftarrow \text{SORT}(\text{EQKP}(t_{\text{query}}, :-1))/\sqrt{d}$ ▷ Cache by t_{query}, i
    $b_{1,:-1} \leftarrow \text{REVERSE}(b_{0,:-1})$
    $a_t \leftarrow \text{EQKE}(t_{\text{query}}, t)/\sqrt{d}$ ▷ Cache by t_{query}, t
    $a_{\text{min},t} \leftarrow \min_{0 \leq t'' \leq t} a_{t''}$ ▷ Cache by t_{query}, t, compute in amortized $\mathcal{O}({d_{\text{vocab}}}^2)$
    $a_{\text{max},t} \leftarrow \max_{0 \leq t'' \leq t} a_{t''}$ ▷ Cache by t_{query}, t, compute in amortized $\mathcal{O}({d_{\text{vocab}}}^2)$
    $\Delta a_{\text{max}} \leftarrow a_{t_{\text{max}}} - a_{\text{min},t_{\text{max}}-g}$ ▷ Cache by t_{query}, t_{max}, c
    $\Delta a_{\text{min}} \leftarrow a_{t_{\text{max}}} - a_{\text{max},t_{\text{max}}-g}$ ▷ Cache by t_{query}, t_{max}, c
    idx-set $\leftarrow \{0, \ldots, n_{\text{ctx}} - c - 1\}$ **if** $t_{\text{max}} \neq t_{\text{query}}$ **else** $\{0, \ldots, n_{\text{ctx}} - c - 2, n_{\text{ctx}} - 1\}$
    attn-weights-unscaled$_{0,i} \leftarrow b_{0,i} + (\Delta a_{\text{min}}$ **if** $i \in$ idx-set **else** $0)$
    attn-weights-unscaled$_{1,i} \leftarrow b_{0,i} + (\Delta a_{\text{max}}$ **if** $i \in$ idx-set **else** $0)$ ▷ Cache by t_{query}, t_{max}, i, c
    attn-weights$_0 \leftarrow$ SOFTMAX(attn-weights-unscaled$_0$) ▷ Cache by t_{query}, t_{max}, i, c
    attn-weights$_1 \leftarrow$ SOFTMAX(attn-weights-unscaled$_1$) ▷ Cache by t_{query}, t_{max}, i, c
    attn-max$_0 \leftarrow \sum_{i \in \text{idx-set}}$ attn-weights$_{0,i}$
    attn-max$_1 \leftarrow \sum_{i \in \text{idx-set}}$ attn-weights$_{1,i}$
    attn-max $\leftarrow$ attn-max$_0$ **if** $\Delta v_{\text{max}}^{t_{\text{max}}} \geq \Delta v_{\text{max}}$ **else** attn-max$_1$
    return skip-score $+ \Delta w_{\text{max,max}} +$ attn-max $\cdot \Delta v_{\text{max}}^{t_{\text{max}}} + (1 -$ attn-max$)\Delta v_{\text{max}}$
end function
function RELAXED-CORRECTNESS-PESSIMIZING-OVER-GAP($\mathcal{M}$, d_{vocab}, n_{ctx}, t_{max}, t_{query}, c, g, g^*)
▷ runs the model on a relaxed variant of input sequences compatible with the arguments
Ensure: CORRECTNESS-PESSIMIZING-OVER-GAP-SLOW is False $\implies$ result is False
Ensure: **return** is False **if** CORRECTNESS-PESSIMIZING-OVER-GAP-SLOW($\mathcal{M}$, $\mathbf{t}$) is False **for** *any* $\mathbf{t}$ with specified t_{max}, t_{query}, and c tokens not equal to t_{max}
    return MODEL-BEHAVIOR-RELAXED-OVER-GAP($\mathcal{M}, t_{\text{max}}, t_{\text{query}}, c, g, g^*) < 0$
end function

G. Subcubic proof strategies

In this section, we present a number of proof strategies that we use to reduce the computational cost of the proof, ultimately driving down the cost of EU and EQKE verification to $\mathcal{O}(d_{\text{vocab}}d_{\text{model}})$, while unfortunately leaving the cost of EVOU verification at $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}})$.

The three main tricks we cover are the mean+diff trick (Appendix G.1), the max row-diff trick (Appendix G.2.2), and the rank one / rank two SVD decomposition of EQKE (Appendix G.2.3). While the mean+diff trick is useful for getting slightly better bounds, the SVD decomposition of EQKE is the place where we get to insert the most understanding (without which we'd have no hope of non-vacuous bounds below $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}})$), and the max row-diff trick is the workhorse that allows us to drive down the noise computations from cubic to quadratic without getting completely vacuous bounds.

G.1. The mean+diff trick

Suppose we have quantities $f_{x,y}$ and $g_{y,z}$ and we want to pessimize (WLOG, suppose minimize) the quantity $f_{x,y} + g_{y,z}$ over x, y, and z in time less than $\mathcal{O}(n_x n_y n_z)$, say we allow $\mathcal{O}(n_x n_y + n_y n_z + n_x n_z)$. Also suppose the variation of f over the y axis is much larger than the variation of f over the x-axis.

We can of course say

$$\min_{x,y} f_{x,y} + \min_{y,z} g_{y,z} \leq f_{x,y} + g_{y,z}$$

But we can do better!

Note that

$$f_{x,y} = \mathbb{E}_x f_{x,y} + (f_{x,y} - \mathbb{E}_x f_{x,y})$$

Suppose that $f_{x,y}$ varies much less over x than it does over y, and much less than $g_{y,z}$ varies over either of y and z. This will make the following bound a good approximation, though the bound is sound even without this assumption. We can write

$$\begin{aligned}
f_{x,y} + g_{y,z} &\geq \min_{x,y,z}[f_{x,y} + g_{y,z}] \\
&= \min_{x,y,z}[\mathbb{E}_x f_{x,y} + g_{y,z} + f_{x,y} - \mathbb{E}_x f_{x,y}] \\
&\geq \min_{x,y,z}[\mathbb{E}_x f_{x,y} + g_{y,z}] + \min_{x,y,z}[f_{x,y} - \mathbb{E}_x f_{x,y}] \\
&= \min_{y,z}[\mathbb{E}_x f_{x,y} + g_{y,z}] + \min_{x,y}[f_{x,y} - \mathbb{E}_x f_{x,y}]
\end{aligned}$$

By averaging the variation over certain axes, we have

Theorem 34 (Mean+Diff)**.**

$$\begin{aligned}
\min_{x,y,z} f_{x,y} + g_{y,z} &\geq \min_{y,z}[\mathbb{E}_x f_{x,y} + g_{y,z}] + \min_{x,y}[f_{x,y} - \mathbb{E}_x f_{x,y}] \\
\max_{x,y,z} f_{x,y} + g_{y,z} &\leq \max_{y,z}[\mathbb{E}_x f_{x,y} + g_{y,z}] + \max_{x,y}[f_{x,y} - \mathbb{E}_x f_{x,y}]
\end{aligned}$$

and the RHSs can be computed in time $\mathcal{O}(n_x n_y + n_y n_z + n_x n_z)$ for n_x, n_y, and n_z the number of possible values of x, y, and z, respectively.

Example for how this helps with small variation:

Take any function $k(y)$ and then take

$$\begin{aligned}
f_{x,y} &:= k(y) + \varepsilon_1(x,y) \\
g_{y,z} &:= -k(y) + \varepsilon_2(y,z)
\end{aligned}$$

Then we have

$$\min_{x,y,z}[f_{x,y} + g_{y,z}] = \min_{x,y,z}[\varepsilon_1(x,y) + \varepsilon_2(y,z)]$$

$$\begin{aligned}\min_{x,y} f_{x,y} + \min_{y,z} g_{y,z} &= \min_y k(y) + \min_y -k(y) + \min_{x,y} \varepsilon_1(x,y) + \min_{y,z} \varepsilon_2(y,z) \\ &= \min_y k(y) - \max_y k(y) + \min_{x,y} \varepsilon_1(x,y) + \min_{y,z} \varepsilon_2(y,z) \\ \min_{x,y}[f_{x,y} - \mathbb{E}_x f_{x,y}] + \min_{y,z}[g_{y,z} + \mathbb{E}_x f_{x,y}] &= \min_{x,y} \varepsilon_1(x,y) + \min_{y,z}[\varepsilon_2(y,z) + \mathbb{E}_x \varepsilon_1(x,y)]\end{aligned}$$

If ε_1 and ε_2 are small compared to $\min_y k(y) - \max_y k(y)$, then using $\mathbb{E}_x f_{x,y}$ gives a much better bound.

Note, though, that this could be a worse bound if the assumption of small variation does not hold.

Note also that this trick is not restricted to adding and subtracting $\mathbb{E}_x f_{x,y}$. If f is a matrix indexed by x and y, we might also try taking SVD and using the first principal component instead. A basic application of the triangle inequality gives the following, more general, result:

Theorem 35 (Summarize+Diff)**.** *For any h_y which can be computed in time $\mathcal{O}(n_h)$,*

$$\begin{aligned}\min_{x,y,z} f_{x,y} + g_{y,z} &\geq \min_{y,z}[h_y + g_{y,z}] + \min_{x,y}[f_{x,y} - h_y] \\ \max_{x,y,z} f_{x,y} + g_{y,z} &\leq \max_{y,z}[h_y + g_{y,z}] + \max_{x,y}[f_{x,y} - h_y]\end{aligned}$$

and the RHSs can be computed in time $\mathcal{O}(n_x n_y + n_y n_z + n_h)$ for n_x, n_y, and n_z the number of possible values of x, y, and z, respectively.

We see that if the variation of f in the x-axis is indeed much smaller than the variation in the y-axis, then letting

$$f_{x,y} = h_y + \varepsilon_{x,y}$$

we get

$$\begin{aligned}&\left|\min_{x,y,z} f_{x,y} + g_{y,z} - \min_{y,z}[h_y + g_{y,z}] - \min_{x,y}[f_{x,y} - h_y]\right| \\ &\leq \left|\min_{x,y,z}[f_{x,y} + g_{y,z}] - \min_{y,z}[h_y + g_{y,z}]\right| + \left|\min_{x,y}[\varepsilon_{x,y}]\right| \\ &\leq 2 \max_{x,y} |\varepsilon_{x,y}|\end{aligned}$$

so indeed this bound isn't too much worse and we are able to compute it in quadratic rather than cubic time.

G.2. Details of SVD of QK proof

As discussed in Section 4.3.1, to further reduce the computation cost of proof, we need to avoid computing the residual stream, EVOU, and EPQKE matrices fully. Using mechanistic insight or otherwise, we observe that these matrices (apart from EVOU) can be well-approximated by rank one matrices. This will remove the dominant computation cost of $\mathcal{O}({d_{\text{vocab}}}^2 \cdot d_{\text{model}})$.

G.2.1. Comments on relationship between mechanistic insight and proof size

Up to this point, we haven't really said much in our proofs about what the model is doing. All the mechanistic insight has been of the form "the model varies more along this axis than this other axis" or "the input data is distributed such that handling these inputs is more important than handling these other inputs" or, at best, "the model computes the answer by attending to the maximum token of the sequence; everything else is noise".

Here, finally, our proof-size constraints are tight enough that we will see something that we could plausibly call "how the model pays attention to the maximum token more than anything else", i.e., (if we squint a bit) "the model pays more attention to larger tokens in general.

G.2.2. The max row-diff trick

As stated above, we are breaking matrices into their rank one approximation and some noise. To bound the noise, i.e. to bound expressions of the form $\prod_i (A_i + E_i) - \prod_i A_i$, where E_i denote the matrix errors, we can use the following trick:

Lemma 36 (Max Row-Diff (vector-matrix version)). For a row vector $\mathbf{a}$ and a matrix B,

$$\max_{i,j}\left((\mathbf{a}B)_i - (\mathbf{a}B)_j\right) \leq \sum_k |a_k| \max_{i,j}\left(B_{k,i} - B_{k,j}\right)$$

Moreover, for a collection of n row vectors A_r, if the shape of B is $m \times p$, the right hand side can be computed for all r in time $\mathcal{O}(nm + mp)$.

Proof.

$$\begin{aligned}
&\max_{i,j}(\mathbf{a}B)_i - (\mathbf{a}B)_j \\
&= \max_{i,j} \sum_k a_k \left(B_{k,i} - B_{k,j}\right) \\
&\leq \sum_k \max_{i,j} a_k \left(B_{k,i} - B_{k,j}\right) \\
&= \sum_k a_k \begin{cases} \max_{i,j}\left(B_{k,i} - B_{k,j}\right) & \text{if } a_k \geq 0 \\ \min_{i,j}\left(B_{k,i} - B_{k,j}\right) & \text{if } a_k < 0 \end{cases} \\
&= \sum_k a_k \begin{cases} \max_{i,j}\left(B_{k,i} - B_{k,j}\right) & \text{if } a_k \geq 0 \\ -\max_{i,j}\left(B_{k,i} - B_{k,j}\right) & \text{if } a_k < 0 \end{cases} \\
&= \sum_k |a_k| \max_{i,j}\left(B_{k,i} - B_{k,j}\right)
\end{aligned}$$

The asymptotic complexity of computing the result follows from caching the computation of $\max_{i,j}\left(B_{k,i} - B_{k,j}\right)$ for each k independently of r, as the computation does not depend on A_r. □

Theorem 37 (Max Row-Diff)**.** *For matrices A and B,*

$$\max_{r,i,j}\left((AB)_{r,i} - (AB)_{r,j}\right) \leq \max_r \sum_k |A_{r,k}| \max_{i,j}\left(B_{k,i} - B_{k,j}\right)$$

Proof. By taking the max of Lemma 36 over rows r of A. □

Lemma 36 can also be applied recursively for a product of more than two matrices.

Lemma 38 (Max Row-Diff (vector-matrix recursive version)). For a row vector $\mathbf{a}$ and a sequence of n matrices B_p of shapes $r_p \times c_p$,

$$\max_{i,j}\left(\left(\mathbf{a}\prod_p B_p\right)_i - \left(\mathbf{a}\prod_p B_p\right)_j\right) \leq \sum_{k_0} |a_{k_0}| \cdots \sum_{k_n} \left|(B_{n-1})_{k_{n-1},k_n}\right| \max_{i,j}\left((B_n)_{k_n,i} - (B_n)_{k_n,j}\right)$$

Moreover, for a collection of q row vectors A_α, the right hand side can be computed for all α in time $\mathcal{O}(qr_0 + \sum_p r_p c_p)$.

Proof. We proceed by induction on n.

For $n = 1$, the statement is identical to Lemma 36.

Suppose the theorem holds for all positive $n = s$; we show the theorem holds for $n = s + 1$. We reassociate the matrix multiplication as

$$\max_{i,j}\left(\left(\mathbf{a}\prod_{p=1}^{s+1} B_p\right)_i - \left(\mathbf{a}\prod_{p=1}^{s+1} B_p\right)_j\right)$$

$$= \max_{i,j} \left((\mathbf{a}B_1) \left(\left(\prod_{p=2}^{s+1} B_p \right)_i - \left(\prod_{p=2}^{s+1} B_p \right)_j \right) \right)$$

Using the induction hypothesis gives

$$\leq \sum_{k_1} \left| \sum_{k_0} a_{k_0} (B_1)_{k_0,k_1} \right| \sum_{k_2} |(B_2)_{k_1,k_2}| \cdots \sum_{k_{s+1}} \left| (B_s)_{k_s,k_{s+1}} \right| \max_{i,j} \left((B_{s+1})_{k_{s+1},i} - (B_{s+1})_{k_{s+1},j} \right)$$

The triangle inequality gives

$$\leq \sum_{k_1} \sum_{k_0} |a_{k_0} (B_1)_{k_0,k_1}| \sum_{k_2} |(B_2)_{k_1,k_2}| \cdots \sum_{k_{s+1}} \left| (B_s)_{k_s,k_{s+1}} \right| \max_{i,j} \left((B_{s+1})_{k_{s+1},i} - (B_{s+1})_{k_{s+1},j} \right)$$

and algebra gives

$$= \sum_{k_0} |a_{k_0}| \sum_{k_1} |(B_1)_{k_0,k_1}| \sum_{k_2} |(B_2)_{k_1,k_2}| \cdots \sum_{k_{s+1}} \left| (B_s)_{k_s,k_{s+1}} \right| \max_{i,j} \left((B_{s+1})_{k_{s+1},i} - (B_{s+1})_{k_{s+1},j} \right)$$

The asymptotic complexity of computing the right hand side also follows straightforwardly by induction. □

Theorem 39 (Max Row-Diff (recursive))**.** *For a sequence of $n+1$ matrices $A_0, \ldots, A_n$,*

$$\max_{r,i,j} \left(\left(\prod_p A_p \right)_{r,i} - \left(\prod_p A_p \right)_{r,j} \right) \leq \max_r \sum_{k_0} |(A_0)_{r,k_0}| \cdots \sum_{k_n} \left| (A_{n-1})_{k_{n-1},k_n} \right| \max_{i,j} \left((A_n)_{k_n,i} - (A_n)_{k_n,j} \right)$$

Proof. By taking the max of Lemma 38 over rows r of A_0. □

Note that Theorem 37 is compatible with the mean+diff trick of Appendix G.1.

Theorem 40 (Combined Mean+Diff and Max Row-Diff)**.** *For matrices A and B, and any column-wise summary vector H_k of A (for example we may take $H_k := \mathbb{E}_r A_{r,k}$)*

$$\max_{r,i,j} \left((AB)_{r,i} - (AB)_{r,j} \right) \leq \left(\max_{i,j} \sum_k H_k \left(B_{k,i} - B_{k,j} \right) \right) + \max_r \sum_k |A_{r,k} - H_k| \max_{i,j} \left(B_{k,i} - B_{k,j} \right)$$

Proof.

$$\begin{aligned}
&\max_{r,i,j} \left((AB)_{r,i} - (AB)_{r,j} \right) \\
&= \max_{r,i,j} \sum_k A_{r,k} \left(B_{k,i} - B_{k,j} \right) \\
&= \max_{r,i,j} \sum_k \left(H_k + (A_{r,k} - H_k) \right) \left(B_{k,i} - B_{k,j} \right) \\
&= \max_{i,j} \left(\sum_k H_k \left(B_{k,i} - B_{k,j} \right) + \max_r \sum_k \left(A_{r,k} - H_k \right) \left(B_{k,i} - B_{k,j} \right) \right) \\
&\leq \left(\max_{i,j} \sum_k H_k \left(B_{k,i} - B_{k,j} \right) \right) + \max_r \sum_k \max_{i,j} \left(A_{r,k} - H_k \right) \left(B_{k,i} - B_{k,j} \right) \\
&\leq \left(\max_{i,j} \sum_k H_k \left(B_{k,i} - B_{k,j} \right) \right) + \max_r \sum_k |A_{r,k} - H_k| \max_{i,j} \left(B_{k,i} - B_{k,j} \right)
\end{aligned}$$

□

Theorem 41 (Combined Mean+Diff and Vector-Matrix Recursive Max Row-Diff)**.** *For a row vector* $\mathbf{a}$*, a vector of summaries* $\mathbf{h}$ *corresponding to* $\mathbf{a}$ *(for example, if* $\mathbf{a}$ *is a row of a matrix,* $\mathbf{h}$ *might be the average of the rows), a sequence of* n *matrices* B_p *of shapes* $r_p \times c_p$*, and a corresponding sequence of column-wise summary vectors* $\mathbf{h_p}$ *of* B_p *(for example we may take* $(h_p)_k := \mathbb{E}_r(B_p)_{r,k}$*),*

$$\begin{aligned}
&\max_{i,j}\left(\left(\mathbf{a}\prod_p B_p\right)_i - \left(\mathbf{a}\prod_p B_p\right)_j\right)\\
&\leq \max_{i,j}\left(\sum_{k_0} h_{k_0}\cdots\sum_{k_n}(B_{n-1})_{k_{n-1},k_n}\left((B_n)_{k_n,i}-(B_n)_{k_n,j}\right)\right)\\
&\quad+\sum_{k_0}|a_{k_0}-h_{k_0}|\cdots\left(\left(\max_{i,j}\sum_{k_n}(h_{n-1})_{k_n}\left((B_n)_{k_n,i}-(B_n)_{k_n,j}\right)\right)\right.\\
&\qquad\qquad\left.+\sum_{k_n}\left|(B_{n-1})_{k_{n-1},k_n}-(h_{n-1})_{k_n}\right|\max_{i,j}\left((B_n)_{k_n,i}-(B_n)_{k_n,j}\right)\right)
\end{aligned}$$

Moreover, for a collection of q *row vectors* A_α*, the right hand side can be computed for all* α *in time* $\mathcal{O}(qr_0+\sum_p r_pc_p)$*.*

Proof sketch. Apply the triangle inequality recursively, fusing the proofs of Lemmas 38 and 40. □

Theorem 42 (Combined Mean+Diff and Recursive Max Row-Diff)**.** *For a sequence of* $n+1$ *matrices* $A_0, \ldots, A_n$*, and corresponding column-wise summary vectors* $\mathbf{h_0}, \ldots, \mathbf{h_{n-1}}$ *of* $A_0, \ldots, A_{n-1}$*,*

$$\begin{aligned}
&\max_{r,i,j}\left(\left(\prod_p A_p\right)_{r,i} - \left(\prod_p A_p\right)_{r,j}\right)\\
&\leq \max_{r,i,j}\left(\sum_{k_0}(h_0)_{k_0}\cdots\sum_{k_n}(A_{n-1})_{k_{n-1},k_n}\left((A_n)_{k_n,i}-(A_n)_{k_n,j}\right)\right)\\
&\quad+\sum_{k_0}|(A_0)_{k_0}-(h_0)_{k_0}|\cdots\left(\left(\max_{i,j}\sum_{k_n}(h_{n-1})_{k_n}\left((A_n)_{k_n,i}-(A_n)_{k_n,j}\right)\right)\right.\\
&\qquad\qquad\left.+\sum_{k_n}\left|(A_{n-1})_{k_{n-1},k_n}-(h_{n-1})_{k_n}\right|\max_{i,j}\left((A_n)_{k_n,i}-(A_n)_{k_n,j}\right)\right)
\end{aligned}$$

Moreover, if the matrices A_p *have shapes* $r_p \times c_p$*, the right hand side can be computed in time* $\mathcal{O}(\sum_p r_pc_p)$*.*

Proof. By taking the max of Theorem 41 over rows r of A_0. □

G.2.3. Exploring rank one approximation via SVD

Let us first look at

$$\text{EQKE} := \mathbf{E}_q Q K^T \bar{\mathbf{E}}^T.$$

From Subfigure 6(a), we see that there is not much variation along long query token direction. We can confirm this by performing a singular value decomposition (SVD) on EQKE, as seen in Figure 11.

The first singular value is just over 7440 (7800 ± 380 across all seeds), while the second singular value is just under 15 (13.1 ± 2.8 across all seeds). The ratio across all seeds is 620 ± 130. There's really not much going on here beyond the first singular component.[19]

[19] We might be tempted to keep analyzing the SVD, and notice that the query direction is mostly uniform, while the key direction is monotonic (nearly linear, even). But the proof complexity doesn't demand this level of analysis, yet, and so we can't expect that any automated compact proof discovery system will give it to us.

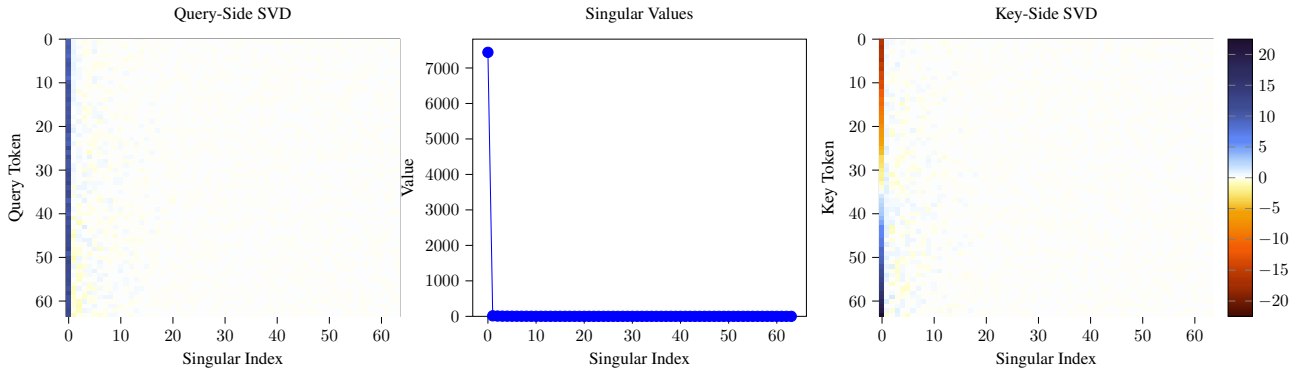


Figure 11: SVD of EQKE for seed 123, with principal component vectors scaled by the square root of the corresponding singular value. This scaling allows us to see visually that there is not much going on beyond the first singular component. Numerically: the first singular value is just over 7440, while the second singular value is just under 15.

Call the first singular component of EQKE the "query direction" d_q and the "size direction" d_k on the query-side and key-side, respectively.

There are two ways that we can decompose EQKE into a low-rank component that we can compute exactly, and a full-rank error term that we approximate bounds for.

G.2.4. The simple SVD decomposition of QK

In time $\mathcal{O}(d_{\text{vocab}}{d_{\text{model}}}^2)$ we can perform SVD on each of the four component matrices $\mathbf{E}_q$, Q, K, $\bar{\mathbf{E}}$ and perform low-rank SVD on the matrix product $\mathbf{E}_q Q K^T \bar{\mathbf{E}}^T$.

We can then bound the difference between two elements in the same row of EQKE by computing exactly the difference between the two elements in the same row of the rank one approximation of EQKE, and adding to that a bound on the difference between the two elements in the same row of the error term.

That is, we can decompose E into a part parallel to d_q and a part orthogonal to d_q, say $\mathbf{E}_q = E_q + E_q^{\perp}$, and similarly $\bar{\mathbf{E}} = E_k + E_k^{\perp}$. Note that E_q and E_k are both rank one, and hence can be multiplied with other matrices of shape $d_{\text{model}} \times a$ in time $\mathcal{O}(d_{\text{model}} a)$ rather than time $\mathcal{O}(d_{\text{vocab}} d_{\text{model}} a)$.

Hence we can define EQKE_err_1 (subscript one for "rank one") and decompose EQKE as

$$\text{EQKE} = E_q Q K^T (E_k)^T + \text{EQKE_err}_1.$$

Define for any vector v

$$\Delta_{i,j} v := v_i - v_j$$

so that we get

$$\begin{aligned} \Delta_{i,j}(E_q Q K^T (E_k)^T)_{t_{\text{query}}} + \min_{i \neq j} \Delta_{i,j}(\text{EQKE_err}_1)_{t_{\text{query}}} \\ \leq \Delta_{i,j} \text{EQKE}_{t_{\text{query}}} \leq \\ \Delta_{i,j}(E_q Q K^T (E_k)^T)_{t_{\text{query}}} + \max_{i \neq j} \Delta_{i,j}(\text{EQKE_err}_1)_{t_{\text{query}}} \end{aligned}$$

Then we may use any method we please to pessimize $\Delta_{i,j}(\text{EQKE_err}_1)_{t_{\text{query}}}$ quickly. For example, since for any matrix M we have $\sigma_1(M) = \sup_x \|Mx\| / \|x\|$, considering vectors with one 1, one -1, and zero elsewhere, the maximum difference between elements in a row upper bounded by $\sqrt{2}\sigma_1(M)$:

$$\left| \Delta_{i,j} \text{EQKE_err}_{1_{t_{\text{query}}}} \right| \leq \sqrt{2}\sigma_1(\text{EQKE_err}_1) \tag{11}$$

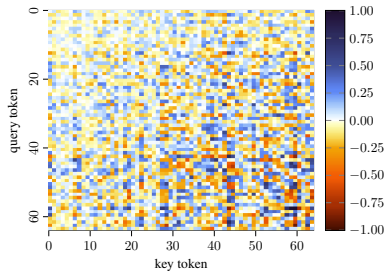


Figure 12: The error term EQKE_err for seed 123.

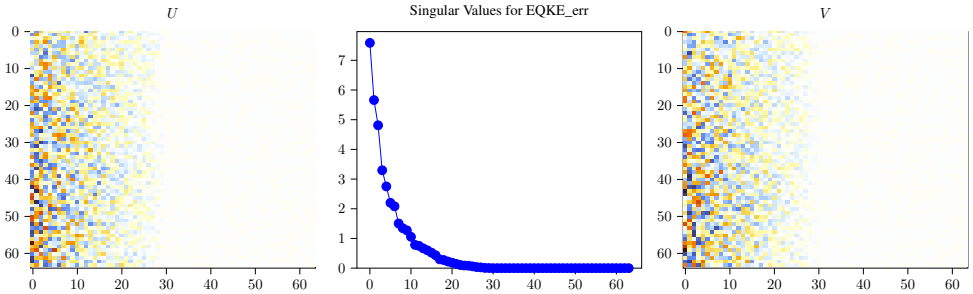


Figure 13: SVD of EQKE_err for seed 123.

G.2.5. The complicated SVD decomposition of QK

While the "most mechanistic" interpretation would proceed with the analysis in terms of E_q and E_k, perhaps decomposing them further, we can get more bang for our buck by extracting out all the low-rank structure available E, Q, and K, so as to make our error bounds as tight as possible.

To this end, we perform SVD on $E_q^\perp$, $E_k^\perp$, Q, and K and peel off the first singular components so as to get the decomposition

$$\begin{aligned}
\mathbf{E}_q &= E_q + E_{q,2} + E_{q,2}^\perp \\
\bar{\mathbf{E}} &= E_k + E_{k,2} + E_{k,2}^\perp \\
Q &= Q_0 + Q^\perp \\
K &= K_0 + K^\perp
\end{aligned}$$

Then EQKE, a product of these four matrices, can be expressed as a sum of $2^2 3^2 - 1 = 35$ rank one products and one high-rank error term. We can compute the sum of the rank one products in time $\mathcal{O}({d_{\text{vocab}}}^2)$ and express EQKE as, say, $\text{EQKE}_2 + E_{q,2}^\perp Q^\perp (E_{k,2}^\perp K^\perp)^T$. Call the second term EQKE_err (Figure 12). We must now bound for each q and m the quantity $\max_{i \le m-G} \text{EQKE_err}[q,i] - \text{EQKE_err}[q,m]$.

How big is this?

Even if we relax to $\max_{i,j} \text{EQKE_err}[q,i] - \text{EQKE_err}[q,j]$, the maximum such value across all rows is under 1.85 (1.99 ± 0.68 across all seeds). And the rows don't have any particular structure to them; the maximum absolute element of the entire matrix is just barely over 1 (1.12 ± 0.40 across all seeds), so doubling that doesn't give too bad an estimate.

But we somehow need to compute this value without multiplying out the four matrices.

One option is to try to use singular value decomposition again. Since $\sigma_1(M) = \sup_x \|Mx\| / \|x\|$, considering vectors with one 1, one -1, and zero elsewhere, the maximum difference between elements in a row upper bounded by $\sqrt{2}\sigma_1(M)$. The largest singular value of EQKE_err (Figure 13) is just under 7.6 (8.4 ± 2.0 across all seeds), giving a row-diff bound of about 10.7 (11.8 ± 2.8 across all seeds), which is large but not unusably so.

If we perform SVD before multiplying out the matrices (Figure 14), however, their first singular values are about 4, 1.4, 1.4, and 4, giving a product of about 30, which when multiplied by $\sqrt{2}$ is about 43. (Across all seeds, these numbers are

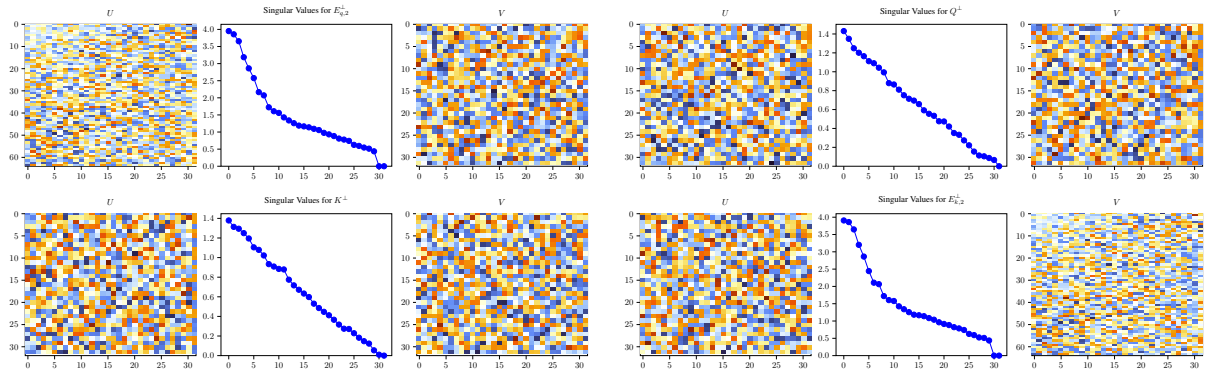


Figure 14: SVD of the four component matrices of EQKE_err for seed 123. Matrices look like noise.

3.79 ± 0.12, 1.525 ± 0.067, 1.513 ± 0.073, and 3.78 ± 0.12, giving a product of about 33.1 ± 2.9, which when multiplied by $\sqrt{2}$ is about 46.8 ± 4.2.) This works because $\sigma_1(AB) \leq \sigma_1(A)\sigma_1(B)$, but note that we can do factored SVD without needing to use this technique. This bound is still usable, but pretty big.

Can we use Frobenius? Note that using anything close to this method to drop below $d_{\text{vocab}}{d_{\text{model}}}^2$ might seem infeasible (it'll eventually turn out not to be). For example, the best bound we know on the largest singular value that can be verified even in the worst-case in strictly less time than it takes to compute the full SVD is the Frobenius norm, which is defined as $\text{tr}(MM^T)$, can be computed in $d_{\text{model}}d_{\text{vocab}}$ time, and is equal to the square root of the sum of the squares of the singular values. While the Frobenius norm of EQKE_err is only about 12 (giving a bound of about 17 on the row-diff), the Frobenius norms of the four multiplicand matrices are a bit over 10, 4, 4, and 10, giving a product of 1932 and a bound of 2732(!). (Across all seeds, the Frobenius norm of EQKE_err is about 13.1 ± 1.9 (giving a bound of about 18.6 ± 2.7 on the row-diff), the Frobenius norms of the four multiplicand matrices are a bit over 9.92 ± 0.19, 4.43 ± 0.01, 4.361 ± 0.095, and 9.85 ± 0.19, giving a product of 1888 ± 99 and a bound of 2670 ± 140.) This is unusably large.

However, we can get a much better bound on the max row-diff of EQKE_err without having to multiply out all four matrices. We can use an approach vaguely similar to the mean+diff trick, as follows.

If we want to compute the max row-diff of a product of matrices AB, we can compute by Theorem 37

$$\max_{r,i,j}((AB)_{r,i} - (AB)_{r,j}) \leq \max_r \sum_k |A_{r,k}| \max_{i,j}(B_{k,i} - B_{k,j}) \tag{12}$$

or by combining this approximation with Theorem 34 via Theorem 40 we may compute

$$\begin{aligned}&\max_{r,i,j}((AB)_{r,i} - (AB)_{r,j}) \\ &\leq \left(\max_{i,j}\sum_k \mathbb{E}_r A_{r,k}(B_{k,i} - B_{k,j})\right) + \max_r \sum_k |A_{r,k} - \mathbb{E}_r A_{r,k}| \max_{i,j}(B_{k,i} - B_{k,j})\end{aligned}$$

taking whichever bound is better.

The first gives us a bound of 7.94 on the maximum row-diff, which is better than we can get by doing SVD on the product of the matrices! We can get an even better bound by peeling off the first two singular values of all four matrices before multiplying them; this gives us a bound of 5.67. Combining it with the avg+diff trick wouldn't give us much (8.05 and 5.66 respectively), as we've effectively already done this by peeling off the leading singular contributions; the mean of EQKE_err over dimension zero has norm 0.025 (0.030 ± 0.012 across all seeds).

Although this noise bound is no longer the leading asymptotic bottleneck, we can peek ahead to what we get if we want to be linear in parameter count. In this case, we can apply the recursive version of Equation 12 via Theorem 39, giving a bound of 97.06 on the maximum row-diff.

The mechanistic understanding we get here is roughly "for any given basis vector of the residual stream, the difference between the overlap of any two input tokens with this direction is small once we factor out the first two singular components",

and this is sufficient to drive a low error term overall if we factor out the leading singular components in other places. We don't mechanistically understand how to combine the $\mathbf{E}_q QK^T$ (without multiplying them out) in a way that allows getting a good bound, though, which corresponds to our inability to drop below $d_{\text{vocab}} {d_{\text{model}}}^2$ here.

If we use this trick on QK only, and use the mean+diff trick on final attention handling (without which we lose about $3\,\%$), we can achieve a bound of 0.7810 (0.797 ± 0.013 across all seeds).

If we use this trick on the skip connection (EU) only, we can achieve a bound of 0.6805 (0.662 ± 0.061 across all seed).

Using this trick on both EU and QK drops us down only to 0.6379 (0.627 ± 0.060 across all seeds).

If we use this trick on EU and use the recursive version of this trick on QK, we get a bound of 0.3064 (0.301 ± 0.036 across all seeds).

Unfortunately, it's not clear how this trick would apply to EVOU. A fancier convex hull checking algorithm seems required, and an analysis thereof is in progress.

H. Comparison of proof strategies

In this section, we compare the various proof strategies that we have developed in Appendix G. We do some traditional mechanistic interpretability analysis to justify that the choices that we made could be expected to lead to reasonably good bounds in Appendix H.1. We then compare the complexities and performance of various proof strategies in Appendix H.2 to line up with the legends of Figures 3 and 4. We close with a figure relating the various categories of proof strategies.

H.1. Justification of pessimization choices

In Sections 4.3, F and G we make a number of choices about which axes of variation are more or less important to track at various points in the bound computation.

Here we do some more traditional mechanistic interpretability analysis to justify that the choices that we made could be expected to lead to reasonably good bounds.

H.1.1. Justifying the gap

We take advantage of the fact that attention is mostly monotonically increasing in input integers and that for most sequences, the attentional contribution of the particular query token matters much more than the particular non-max token in the sequence.

We justify this as follows.

We can look at the typical diff, when attending to the max token, between the largest non-max logit and the max logit. As shown in Subfigure 15(a), the largest difference between an off-diagonal entry of EVOU and the diagonal of that row is typically at most -7.[20] The typical worst contribution to the wrong logit from a non-max token (this is typical over non-max tokens, worst over choice of output token-logit index) is around 43, as shown in Subfigure 15(b).

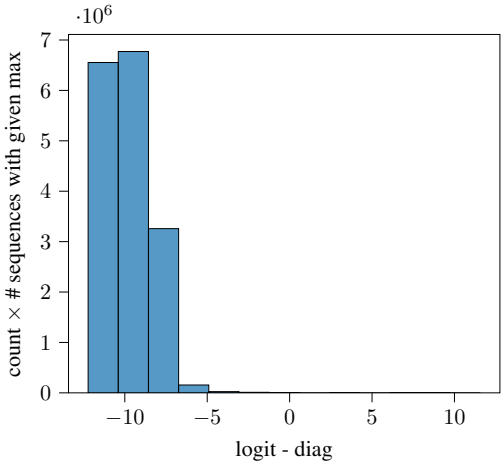
The difference in attention between tokens is approximately linear in the gap between the tokens, as seen in Figure 16. The slope of the line, that is, the difference in pre-softmax attention scores divided by the gap between the key token and the max token, is approximately 1.2.

Exponentiating, the post-softmax attention paid to the max is typically about $3\times$ larger than to the token one below the max; here the logit difference between the max and non-max token is significant, typically being around 13 ($43/3$) for the worst output logit. But by the time the gap is 3, this difference has dropped to about 1.1, and by the time the gap is 4 it is around 0.3.

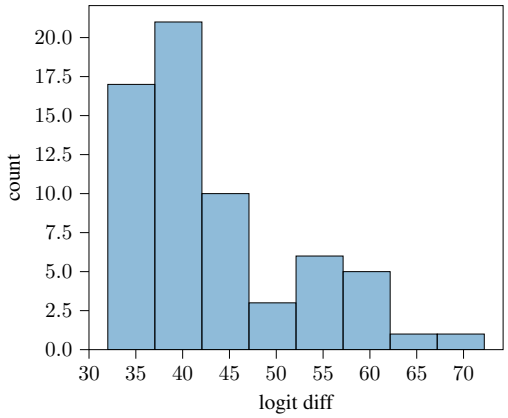
So for sequences where the largest non-max and the max are close together, the particular structure of the non-max EVOU matters a lot; but when the max is separated from the largest non-max by a modest gap, the structure of the non-max EVOU does not matter so much.

The upshot is that to handle most sequences, we need only ask an oracle for the minimum gap $g > 0$ between the max token

[20] "Typically" here means about $96\,\%$ of the time.

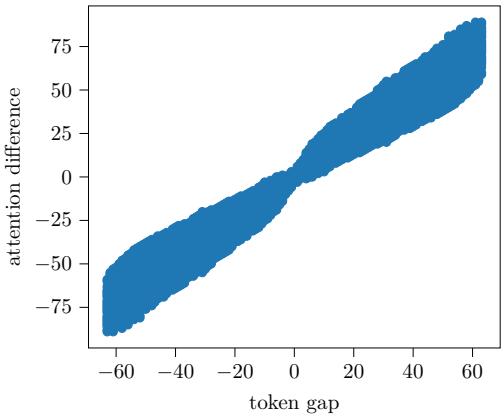


(a) The attention computation weighted by the number of sequences with the particular max. The computation is $\max_{j \text{ s.t. } j \neq i} \mathrm{EVOU}_{i,j} - \mathrm{EVOU}_{i,i}$. $\mu \pm \sigma$: -9.9 ± 2.1; range: -0 ± 12

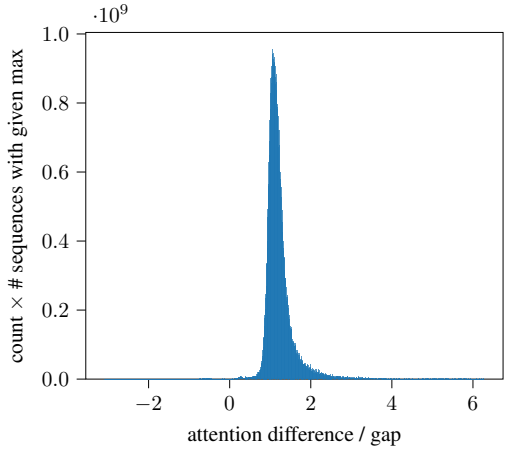


(b) Histogram of the maximum difference between two logits contributed by a single row of EVOU. The computation is, for each i, $\max_h \mathrm{EVOU}_{i,j} - \min_j \mathrm{EVOU}_{i,j}$. $\mu \pm \sigma$: 43.4 ± 9.0; range: 52 ± 20

Figure 15: Plots of the difference in logit for the attention computation, $\mathrm{EVOU} := \bar{\mathbf{E}}VOU$ for seed 123.



(a) $(\mathrm{EQKE}_i - \mathrm{EQKE}_j)/\sqrt{d}$ vs. $i - j$



(b) $(\mathrm{EQKE}_i - \mathrm{EQKE}_j)/(\sqrt{d}(i-j))$, weighted by sequence count. $\mu \pm \sigma = 1.22 \pm 0.13$

Figure 16: Plots of attention difference vs. token gap, for $\mathrm{EQKE} := \mathbf{E}_q Q K^T \bar{\mathbf{E}}^T$ for seed 123. The difference in attention between tokens is approximately linear in the gap between the tokens.

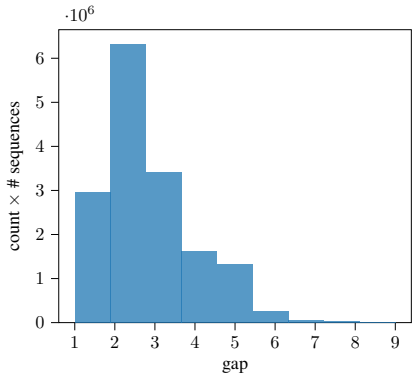


Figure 17: Histogram of the minimum gap between the max token and the largest non-max token, for the seed 123.

$t_{\max}$ and largest non-max tokens $t' \neq t_{\max}$, such that the model outputs the correct answer for all sequences where the non-max, non-query tokens have value at most $t_{\max} - g$.

While computing this gap may be expensive (and indeed the naïve computation of the oracle takes longer than the brute-force proof—though it should be very easy to optimize), we don't have to pay the cost of computing the gap in the size of the proof, only the cost of storing the gap table ($\mathcal{O}({d_{\text{vocab}}}^2 n_{\text{ctx}})$) and of verifying the gap. Empirically, gaps are typically 1–6, as seen in Figure 17.

If we rely on the gaps, this results in leaving behind about $7.9\,\%$ of sequences.

Picking up more sequences In this paragraph / bulleted list, we sketch out how we might go about picking up more sequences to get a tighter bound. This is not coded up, and is left as future work. We propose computing the following quantities:

- First, we could build in time ($\mathcal{O}({d_{\text{vocab}}}^2)$) a table indexed on pairs $(t, t_{\max})$ of the maximum token and a non-maximum token: the table would store pessimal logit contributions from t to maximum output tokens $\leq$ the $t_{\max}$ parameter. The table could be further split to pessimize separately for tokens within and outside of the gap window.
- Compute a table of pre-softmax attention differences between tokens t and $t+1$ in time ($\mathcal{O}({d_{\text{vocab}}}^2)$).
- Next sort the queries by overlap with the query direction.
- Compute for each number of queries handled (where we assume we handle all queries with greater overlap than the current one) and for each maximum input token $t_{\max}$, how many of the query tokens t_{query} fall strictly below the max $t_{\max}$ (and whether or not the model succeeds when $t_{\max} = t_{\text{query}}$). This will tell us how many query tokens we can count for a given maximum token.
- Compute a table indexed on pairs of # of queries handled and input tokens t which stores the smallest difference in more attention paid to $t+1$ than to t ($\mathcal{O}({d_{\text{vocab}}}^2)$).
- Compute a table indexed on pairs $t_{\max}$, t storing an upper bound on amount more attention paid to non-maximum tokens than to $t_{\max}$ by Oracle-permitted query tokens (the Oracle is indexed only on $t_{\max}$) ($\mathcal{O}({d_{\text{vocab}}}^2)$).
- For each # queries permitted: compute for each $t_{\max}$, t, c, if the non-maximum token t contributes little enough to incorrect logits that even with the worst skip connection the model still gets the correct answer.

H.1.2. Stopping after 1–2 principal components of QK

Did we miss out on any structure in the noise of EQKE? The distribution of entries of the four matrices looks pretty close to normal as seen in Figure 18.

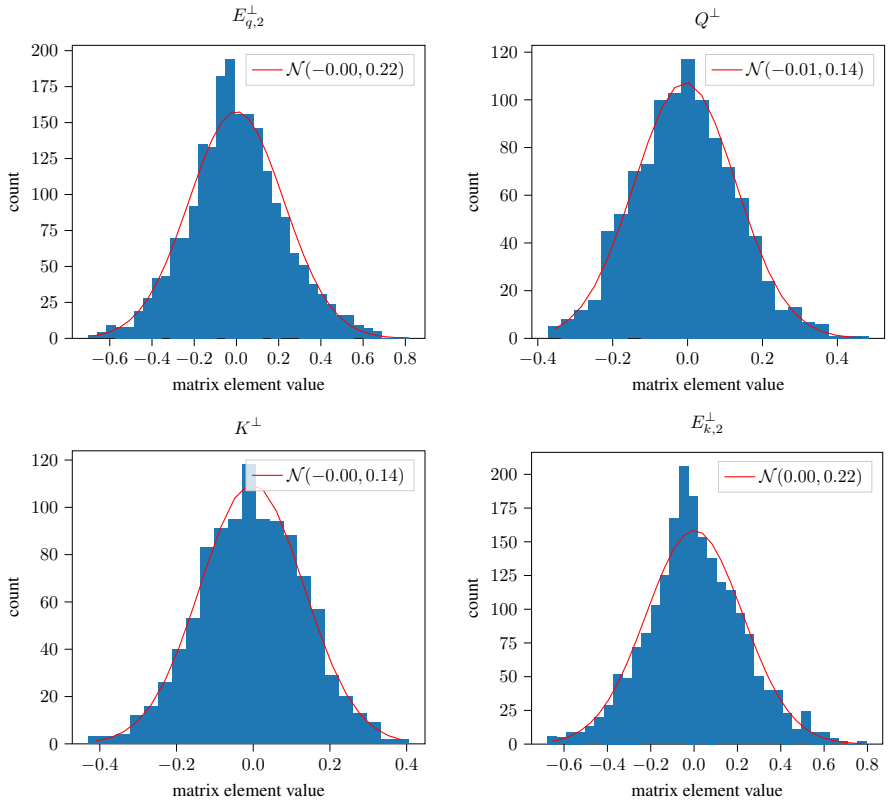


Figure 18: The distribution of entries of the four residual matrices (after removing two principal components from $\mathbf{E}_q$ and $\bar{\mathbf{E}}$ and one principal component from Q and K). Distributions look pretty close to normal. Plots are for the seed 123.

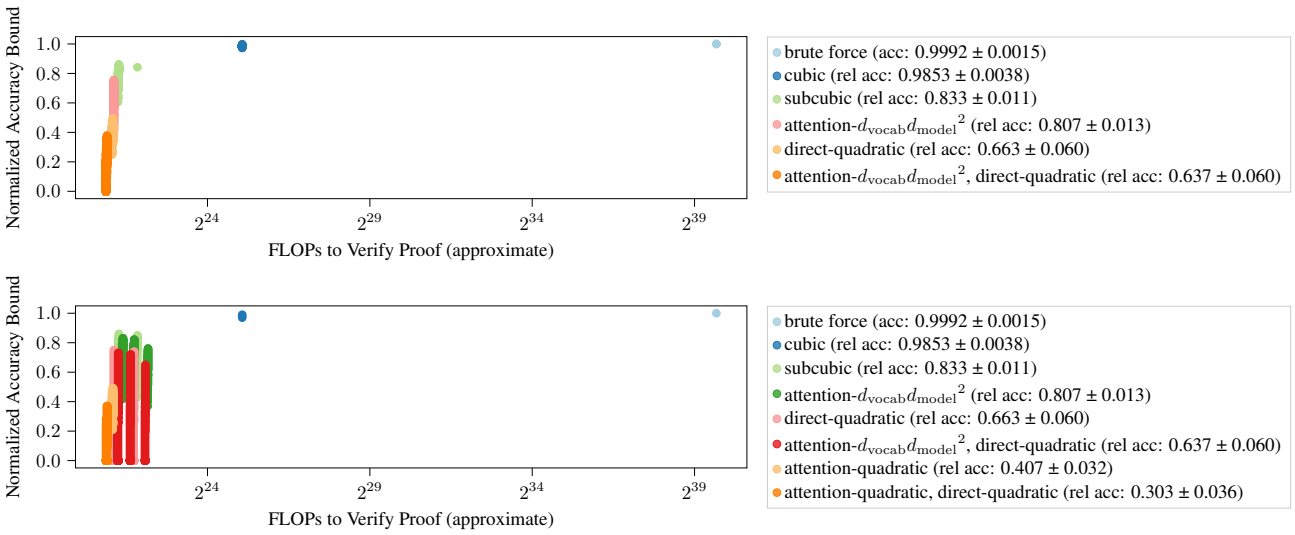


Figure 19: Recreations of Figure 3 for ease of viewing of the legend. Top is a strict recreation; bottom includes points not on the Pareto frontier.

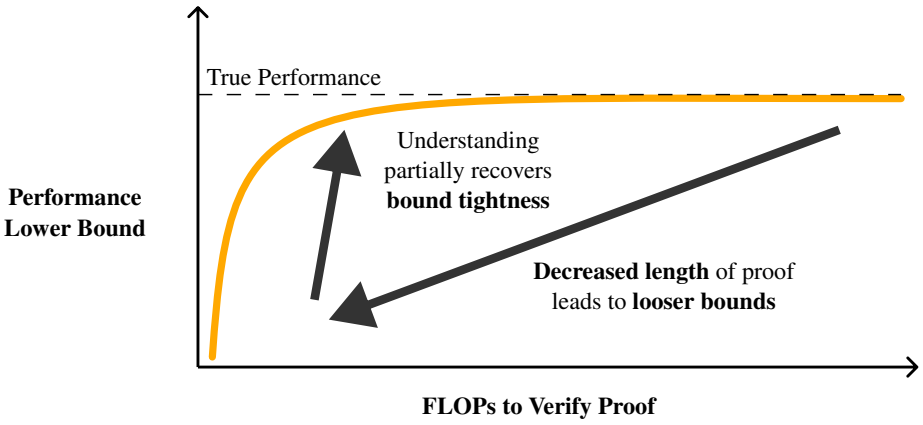


Figure 20: Shorter proofs by default have a worse performance bound b. Faithful understanding allows us to recover significant bound tightness with minimal proof-length overhead.

If we sample elements randomly, we get (sample size 100) that the maximum row-diff of the product of the matrices is approximately 1.31 ± 0.13 (sampling without replacement from the empirical distribution) or 1.31 ± 0.14 (sampling from the normal distribution). So in fact our max row-diff is unusually high (by about 4σ).[21]

H.2. How various combinations of tricks perform

Recall Figures 3 and 4 on page 8 from Section 5, recapitulated here without captions for convenience as Figures 19 and 21.

We describe what each subcubic proof strategy in the legend means. Note that all subcubic proof strategies (that is, all proof strategies except for "brute force" and "cubic") use the quadratic counting algorithm of Appendix F.

[21] This shows up in the bias towards having larger values (both positive and negative) in the lower-right corner of the plot, indicating that noise is larger for larger query and key values. We hypothesize that this is due to the distribution of data: larger values are more likely to have more space between the maximum and next-most-maximum token, so a bit of noise matters less for larger maxes than for smaller ones.

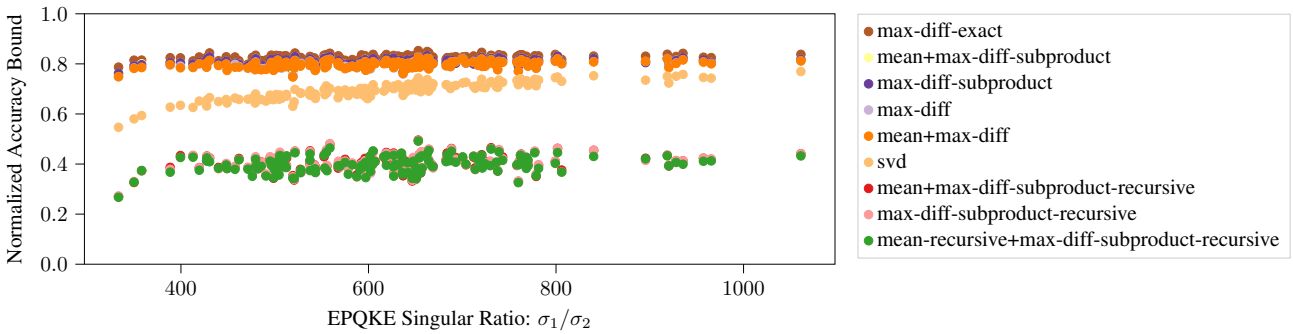


Figure 21: Recreation of Figure 4 for ease of viewing of the legend.

H.2.1. Proof strategies grouped by complexity

In Figures 3 and 19, proof strategies are grouped by computational complexity.

The 102 proof strategies break down into $1 + 1 + 2 \times 5 \times 10 \times 2$ strategies.

The **brute force** and **cubic** proofs $(1 + 1)$ were fully covered in Appendices D and E.

There are 5 options for handling EU:

direct-quadratic refers to handling EU in time $\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$ with either the max row-diff trick (Appendix G.2.2)[22] or the max row-diff trick fused with mean+diff or some other summary statistic (Theorem 40)[23].

When **direct** is not mentioned, this indicates that we handle EU in time $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}})$ by first multiplying out $\mathbf{E}_q U$ and then either taking the maximum row-diff in each row[24] or by taking the maximum row-diff across all rows[25]. The latter is included purely for comparison's sake, and never gives a tighter bound than the former.

There are 10 options for handling the high-rank attention error term EQKE_err:

attention-quadratic refers to handling the high-rank attention error term EQKE_err from Appendix G.2.5 in time $\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$ either with the recursive max row-diff trick (Theorem 39)[26] or with the recursive max row-diff trick fused with the mean+diff trick either just on the query side[27] or throughout[28] (Theorem 42).

attention-$d_{\text{vocab}} {d_{\text{model}}}^2$ indicates that we use one of the various $\mathcal{O}(d_{\text{vocab}} {d_{\text{model}}}^2)$ strategies for handling EQKE_err_1 from Appendix G.2.4 or EQKE_err from Appendix G.2.5. These include using $\sqrt{2}\sigma_1$—computed via low-rank SVD—as the bound (Equation 11)[29], considering all ways of multiplying out a subset of the matrices and taking the maximum row-diff of the resulting pair of matrices[30] (Theorem 37), or fusing the max row-diff trick with the mean+diff trick[31] (Theorem 40).

When **attention is not mentioned**, this indicates that we handle the attention error term in time $\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}})$, either by taking the per-row maximum row-diff[32] or by using the full rank EQKE matrix and taking the per-row maximum row diff[33].

Finally, note that in combining the rank one attention computation with EVOU, PVOU, and EU, we may either use the

[22] This strategy is labeled "`max_diff`" in the Python source code.
[23] These strategies are labeled "`mean_query+max_diff`" and "`svd_query+max_diff`" in the Python source code
[24] "`max_diff_exact`"
[25] "`global_max_diff_exact`"
[26] "`max_diff_subproduct_recursive`"
[27] "`mean+max_diff_subproduct_recursive`"
[28] "`mean_recursive+max_diff_subproduct_recursive`"
[29] "`svd`"
[30] "`max_diff`" for EQKE_err_1, "`max_diff_subproduct`" for EQKE_err
[31] "`mean+max_diff`" for EQKE_err_1, "`mean+max_diff_subproduct`" for EQKE_err
[32] "`max_diff_exact`"
[33] "`exact_EQKE+max_diff_exact`"

mean+diff trick[34] (Appendix G.1) or not[35]; this makes up the final factor of 2.

H.2.2. Proof strategies grouped by attention handling

This section slightly reorganizes the information just covered in Appendix H.2.1, for convenience of legend correspondence. Here we group by the strategy used to handle the attention error term. Strategies that involve using the full rank EQKE matrix are elided. The dashed descriptors here correspond to underscore-joined descriptors in footnotes of Appendix H.2.1.

max-diff-exact ($\mathcal{O}({d_{\text{vocab}}}^2 d_{\text{model}})$) corresponds to taking the full rank EQKE_err_1 term and taking the maximum row-diff in each row.

mean+max-diff-subproduct ($\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$) corresponds to fusing the max row-diff trick with the mean+diff trick (Theorem 40) and considering all ways of associating the multiplication of EQKE_err.

max-diff-subproduct ($\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$) corresponds to using the max row-diff trick (Theorem 37) and considering all ways of associating the multiplication of EQKE_err.

max-diff ($\mathcal{O}(d_{\text{vocab}} {d_{\text{model}}}^2)$) corresponds to using the max row-diff trick (Theorem 37) on the factored SVD of EQKE_err_1.

mean+max-diff ($\mathcal{O}(d_{\text{vocab}} {d_{\text{model}}}^2)$) corresponds to fusing the max row-diff trick with the mean+diff trick (Theorem 40) and applying it on the factored SVD of EQKE_err_1.

svd ($\mathcal{O}(d_{\text{vocab}} {d_{\text{model}}}^2)$) corresponds to using $\sqrt{2}\sigma_1$—computed via low-rank SVD—as the bound (Equation 11).

mean+max-diff-subproduct-recursive ($\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$) corresponds to handling the high-rank attention error term EQKE_err from Appendix G.2.5 with the recursive max row-diff trick fused with the mean+diff trick on the query-side only (Theorem 42, taking all but the first summary vector to be zero).

max-diff-subproduct-recursive ($\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$) corresponds to handling the high-rank attention error term EQKE_err from Appendix G.2.5 with the recursive max row-diff trick (Theorem 39).

mean-recursive+max-diff-subproduct-recursive ($\mathcal{O}(d_{\text{vocab}} d_{\text{model}})$) corresponds to handling the high-rank attention error term EQKE_err from Appendix G.2.5 with the recursive max row-diff trick recursively fused with the mean+diff trick (Theorem 42).

H.2.3. What understanding do we get from each proof strategy?

Throughout most of this paper, we talk about doing mechanistic interpretability and using understanding to allow more compact proofs to have tighter bounds. We can also look at the reverse problem: we can take a collection of proof strategies, check by brute force which strategies give the tightest bounds for each model, and ask what this implies about how that model works. We do this here.

In general, which proof methods perform best is an indication of where structure exists in the model. For example, in quadratic EU proofs, when `max_diff` performs worse than `mean_query+max_diff` and `svd_query+max_diff`, this indicates that E has a relatively strong behavioral component shared across query tokens that U is not that good at filtering out. Similarly, when, e.g., `mean_recursive+max_diff_subproduct_recursive` performs better than `max_diff_subproduct_recursive`, this indicates that even after removing the first one or two principle components from $\mathbf{E}_q$, Q, K, and $\bar{\mathbf{E}}$, there is still enough common structure that it is worth factoring out the mean behavior.

[34] "`mean_query+diff`"

[35] "`drop_average_query_per_output_logit_reasoning`"