Fixed Universal Transformers

Jingwen Liu
Columbia University
New York, NY
jingwenliu@cs.columbia.edu

Alexandr Andoni
Columbia University
New York, NY
andoni@cs.columbia.edu

Daniel Hsu
Columbia University
New York, NY
djhsu@cs.columbia.edu

Abstract

We introduce *universal transformers*: fixed transformers that can simulate any transformer in a given class via a suitable input embedding. Analogous to a universal Turing machine, the input embedding encodes a description of the target model while all internal parameters remain fixed. We provide explicit sparse constructions achieving universality when the embedding dimension is sufficiently large, and further show that universality is generic: randomly initialized transformers are universal almost surely, which aligns with recent empirical results of Zhong and Andreas (2024). We empirically validate our theory on the algorithmic tasks of parenthesis balancing and multi-hop reasoning. Our results suggest that much of a transformer's expressive power may reside in its input representation rather than its learned weights.

1 Introduction

We introduce the notion of a *universal transformer*: a fixed transformer that can simulate any transformer in a prescribed class by changing only the input embedding. In a universal transformer, the embedding plays the role of a program: it encodes the parameters of the target transformer, while the attention and value parameters of the universal transformer remain fixed. This notion can be regarded as a deep learning analogue of the concept of universality from the theory of computation, of which universal Turing machines [23] and universal circuits [24] are the prime examples. A universal Turing machine is a single Turing machine that is capable of simulating any other Turing machine (provided in a suitable encoding) as input, much like how we regard general purpose computers today [23]. A universal circuit is a single circuit that can simulate any target circuit of comparable size by simply fixing additional special "control" inputs to an appropriate values specific to the target circuit [24]. Note that universality here is not the same as universal approximation, which refers to the ability to choose the model parameters to approximate an arbitrary function. Instead, the architecture and non-embedding parameters are fixed, and the question is, for any target transformer of comparable form, whether there is a suitable representation of the input that makes this fixed computation emulate the target transformer.

Our notion of universality is partly motivated by recent experimental findings of [29] on *random transformers*, where they found that many basic capabilities of language models are already present in randomly-initialized transformers, with only trained embedding and unembedding layers. These random transformers turn out to be universal (almost surely). Moreover, randomness is not essential: universal transformers can also be *explicitly* constructed. Our constructions give more insight into how the emulation can be carried out, and they do not resemble (and are much sparser than) any

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

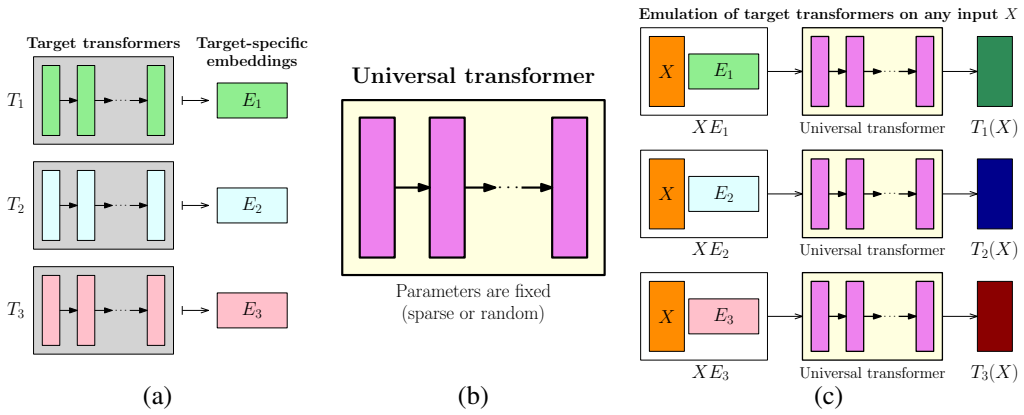


Figure 1: Schematic overview of universal transformers. (a) Each target transformer corresponds to a target-specific embedding (based on the target's model parameters). (b) The parameters of the universal transformer are fixed, either using the sparse deterministic construction from Theorem 3.1, or chosen randomly as in Theorem 3.3. (c) The composition of the universal transformer with the target-specific embedding emulates the target transformer on all inputs.

commonly-used initialization scheme for transformers. Universal transformers are also related to model reprogramming, a transfer-learning paradigm that adapts a pre-trained model to new domains by changing only the embedding layers [2]; the existence of universal transformers vastly expands the potential scope of that approach.

Our contributions. We formalize the notion of a universal transformer, which can simulate any transformer within a given class via a suitable input embedding. We provide explicit constructions of such models and further show that randomly initialized transformers are universal.

- **Explicit sparse universal transformers.** We show that there exists a fixed H-head, L-layer universal transformer with embedding dimension m and $\{0,1\}$-valued parameters that can simulate any H-head, L-layer transformer with embedding dimension d, provided that $m = O(Ld)$ when $H = 1$ and $m = O(H^L d)$ when $H \geq 2$. Moreover, this result extends to the looped (weight-tied) setting, where the same parameters are reused across layers in the universal transformer.

  Theorem 1.1 (Informal version of Theorems 3.1 and 3.2)**.** *There exists fixed $\{0,1\}$-valued parameters defining a H-head L-layer transformer (or looped transformer)* UT *with embedding dimension $m = O(Ld)$ when $H = 1$ and $m = O(H^L d)$ when $H \geq 2$, such that there are at most m nonzero entries in each parameter matrix, and for any H-head L-layer transformer with embedding dimension d, there exists an embedding matrix $E \in \mathbb{R}^{d \times m}$, we have $\mathrm{UT}(XE) = T(X)$ for any input X.*

- **Random universal transformers.** We prove that universality is in abundance: almost surely, a randomly initialized H-head, L-layer transformer can simulate any H-head, L-layer transformer with embedding dimension d, when the embedding dimension satisfies $m = O(H^L d)$.

  Theorem 1.2 (Informal version of Theorems 3.3 and 3.4)**.** *Draw random parameters from an absolutely continuous distribution. With probability 1, the resulting H-head L-layer transformer (or looped transformer)* UT *with embedding dimension $m = O(H^L d)$ such that for any H-head L-layer transformer with embedding dimension d, there exists an embedding matrix $E \in \mathbb{R}^{d \times m}$, we have $\mathrm{UT}(XE) = T(X)$ for any input X.*

- **Lower bounds.** We establish that, when the number of heads is $H \geq 2$, the exponential dependence on depth L is unavoidable, even when the embedding dimension is 1, for any approach based on "attention-matching", where the universal transformer reproduces the attention matrices of the transformer being simulated.

  Theorem 1.3 (Informal version of Theorem 4.1)**.** *If an H-head, L-layer transformer* UT *with embedding dimension m can simulate any H-head, L-layer transformer T with embedding dimension 1 via "attention-matching", then $m \geq H^{L/2}$.*

- **Empirical evaluation.** We evaluate sparse and random universal transformers on parenthesis balancing and k-hop induction head tasks, achieving perfect or near-perfect accuracy. These findings complement our representational results and demonstrate that standard gradient-based optimization can effectively learn the required embeddings.

1.1 Related work

Universality in computation. Our notion of a universal transformer takes inspiration from classical notions of universality in the theory of computation [23, 24], but adapts it to the deep learning context. Specifically, the description of the target transformer is provided as an input embedding layer for the universal transformer.

As mentioned in the introduction, universal transformers can also be regarded as a form of model reprogramming [2]. Theoretical analyses on model reprogramming rely on distributional assumptions and offer average case error guarantees [26], much like other works on transfer learning. Our results offer a complementary type of theoretical guarantee for model reprogramming when the "pre-trained model" is a universal transformer, and the target tasks are solvable using task-specific transformers.

Universal approximation and Turing-completeness. The notion of universality we study in this work is distinct from the universal approximation property of various types of neural networks (including transformers) [e.g., 6, 13, 11, 28, 25], which is concerned with the ability to arbitrarily-well approximate broad classes of functions by entire families of neural networks. In particular, for any given target function, the approximating network's size and internal parameters may all be specific to the target function. Similarly, many network families have been shown to satisfy Turing-completeness, which (in this context) means that any Turing machine can be simulated by an appropriate member of the network family [22, 7, 5, 25, 12]. (We have an unfortunate clash of terminology with [7], who use the term "Universal Transformer" for a recurrent extension of the standard transformer architecture that the authors argue is Turing-complete; our work does not change the transformer architecture.) Again, the particular network that simulates a given Turing machine may have size and internal parameters that depend on that Turing machine.

The Turing-completeness of transformers offers one avenue to construct universal transformers: use the fact that every transformer can be simulated by a Turing machine (e.g., one that implements a Python interpreter with PyTorch) when provided in a suitable encoding (e.g., model `.pt` file), and then obtain a transformer that simulates *that* Turing machine. However, such an approach would not (at least readily) come with any efficiency or succinctness guarantees about the constructed universal transformer, even for the classes of transformers we consider in our results.

Random networks. Random networks have long served as a lens for studying the capabilities of a wide range of machine learning models (especially at random initialization), including kernel machines [17, 20], feedforward neural networks [14, 10], recurrent neural networks [15, 16, 1], and—more recently—transformers [3, 29, 8, 19]. As mentioned in the introduction, our work is, in part, inspired by the empirical study of [29] on random transformers, which tested and validated the hypothesis that many capabilities of language models are already present in randomly-initialized transformers, prior to any training. That work also empirically observed that the token representations at different levels of a random transformers seem to lie close to low-dimensional subspaces corresponding to the current task at hand, but also that these representations could not be sparsified by selecting individual neurons in the multi-layer Perceptron (MLP) layers; this is taken as evidence against a "lottery ticket hypothesis" [10, 19] for the success of random transformers.

The work of [8] continues the investigation of random transformers by further empirically investigating the relative importance of the self-attention parameters and MLP parameters. Theoretically, they attribute the expressivity of "Frozen-QK" transformers—where only the self-attention parameters are fixed at their initialization, but the MLP parameters are trained—to the universal approximation capabilities of MLPs and an embedding dimension that is sufficiently large for self-attention to gather the entire input sequence into a single token embedding. This explanation for the capabilities of (a different type of) random transformers is complementary to that provided by universality. Our theoretical work focuses on attention-only transformers (as opposed to transformers with MLP layers), considers exact representation (as opposed to approximation), and does not use any target-specific layers besides the embedding layers.

As already pointed out by [29], the lottery ticket hypothesis concerns a rather different property of random networks that involves finding useful subnetworks through sparsification/pruning. This hypothesis has been explored both empirically [3] and theoretically [19] for transformers, but it is not a basis for the universality property that we study.

2 Preliminaries

Notation. For a positive integer k, we denote $[k] := \{1, 2, \ldots, k\}$. We use $\{0,1\}^{m\times n}$ to denote the class of $m \times n$ matrices with entries in $\{0, 1\}$. For a matrix $X \in \mathbb{R}^{n\times d}$, we interpret the n rows as tokens, where n is the context length, and d as the feature dimension. We use $\mathrm{softmax}$ to denote the row-wise softmax function.

2.1 Standard transformer

We consider a standard multi-head self-attention transformer without MLP layers.

Transformer class. We use $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$ to denote the class of H-head L-layer transformers, with the input and output dimension d_{in}, and each head uses an internal head dimension d.

A transformer $T \in \mathrm{TF}_{H,L,d_{\mathrm{in}},d}$ is defined by parameters including query and key matrices $W_Q^{l,h}, W_K^{l,h} \in \mathbb{R}^{d_{\mathrm{in}}\times d}$, value matrix $W_V^{l,h} \in \mathbb{R}^{d_{\mathrm{in}}\times d}$, and output matrix $W_O^{l,h} \in \mathbb{R}^{d\times d_{\mathrm{in}}}$, for each layer $l \in [L]$ and head $h \in [H]$, and the following computation:

Given an input sequence $X^0 \in \mathbb{R}^{n\times d_{\mathrm{in}}}$, T computes for each layer $l \in [L]$:

$$A^{l,h} := \mathrm{softmax}\Big((X^{l-1}W_Q^{l,h})(X^{l-1}W_K^{l,h})^\top\Big) \in \mathbb{R}^{n\times n},$$

$$X^l := \sum_{h=1}^{H} A^{l,h}X^{l-1}W_V^{l,h}W_O^{l,h} \in \mathbb{R}^{n\times d_{\mathrm{in}}}.$$

The final output is $T(X) := X^L \in \mathbb{R}^{n\times d_{\mathrm{in}}}$.

2.2 Universal transformer

We now define the *universal transformer* model. Like universal circuits, we define a universal transformer with respect to a class of transformers of certain set of hyper-parameters.

Universal transformer model. We define a parametric model $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ which is an H-head, L-layer transformer with embedding dimension m. It is parameterized by matrices $R_Q^{l,h}, R_K^{l,h} \in \mathbb{R}^{m\times d}$ and $R_V^{l,h} \in \mathbb{R}^{m\times m}$ for each $(l, h) \in [L] \times [H]$, together with an unembedding matrix $U \in \mathbb{R}^{m\times d_{\mathrm{in}}}$.

Given an input sequence $X^0 \in \mathbb{R}^{n\times d_{\mathrm{in}}}$ and an embedding matrix $E \in \mathbb{R}^{d_{\mathrm{in}}\times m}$, $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ first maps the input into the m-dimensional space:

$$\tilde{X}^0 := X^0E \in \mathbb{R}^{n\times m}.$$

For each layer $l \in [L]$, it then computes

$$\tilde{A}^{l,h} := \mathrm{softmax}\Big((\tilde{X}^{l-1}R_Q^{l,h})(\tilde{X}^{l-1}R_K^{l,h})^\top\Big) \in \mathbb{R}^{n\times n},$$

$$\tilde{X}^l := \sum_{h=1}^{H} \tilde{A}^{l,h}\tilde{X}^{l-1}R_V^{l,h} \in \mathbb{R}^{n\times m}.$$

Finally, the output is projected back to the input space:

$$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}(X, E) := \tilde{X}^L U \in \mathbb{R}^{n\times d_{\mathrm{in}}}.$$

The key property of the universal transformer is that it can simulate any transformer in $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$ via a suitable choice of embedding.

Definition 2.1 (Universal transformer)**.** *We call a* $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$*, parametrized by fixed* $\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}, U\}$*, a* universal transformer *for the class* $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$ *if, for any* $T \in \mathrm{TF}_{H,L,d_{\mathrm{in}},d}$*, there exists an embedding matrix* E *such that*

$$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}(X, E) = T(X) \quad \textit{for all } X \in \mathbb{R}^{n \times d_{\mathrm{in}}} \textit{ and all } n \in \mathbb{N}.$$

Note that the universal transformer must be *uniform over the context length* n: the parameters $\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}, U\}$ are fixed and do not depend on n, and the same universal transformer can be applied to inputs of arbitrary length.

This concept is analogous to concepts of universal Turing machines [23] and universal circuits of [24], adapted to the deep learning context. In our setting, the embedding matrix E serves as a *description* (or encoding) of the target transformer being simulated, while the parameters $\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}, U\}$ define a fixed computational template that performs the simulation. The embedding dimension m therefore quantifies the expressive capacity of the universal transformer, i.e., the amount of information needed to encode a target transformer within this fixed architecture. In Sections 3 and 4, we study the minimal m required for universality over the class $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$.

We also consider a *looped* (or weight-tied) variant of the universal transformer, in which the same attention parameters are reused across iterations.

Looped universal transformer model. We define a looped parametric model looped-$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ which is an H-head transformer unrolled for L iterations, with embedding dimension m. It is parameterized by shared matrices $R_Q^h, R_K^h \in \mathbb{R}^{m \times d}$ and $R_V^h \in \mathbb{R}^{m \times m}$ for each $h \in [H]$, together with an unembedding matrix $U \in \mathbb{R}^{m \times d_{\mathrm{in}}}$. The looped-$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ follows the same computation path as $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ but replacing $R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}$ with R_Q^h, R_K^h, R_V^h.

3 Constructing universal transformers

In this section we characterize when a transformer can serve as a *universal transformer* for the class $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$, and to determine the minimal embedding dimension m required for universality.

3.1 Fixed sparse universal transformer

In the following theorem, we give an explicit construction of a *sparse* universal transformer, which establishes an upper bound on the required embedding dimension and serves as the basis for our experiments. The upper bound is given in terms of the following quantity $\overline{m}$:

$$\overline{m} := \begin{cases} (L+1)\max\{2d, d_{\mathrm{in}}\}, & \text{if } H = 1, \\ \frac{2H(H^L-1)}{H-1} d + H^L d_{\mathrm{in}}, & \text{if } H > 1. \end{cases}$$

Theorem 3.1 (Sparse universal transformer)**.** *There exist fixed parameters* $\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}\}_{l \in [L],\, h \in [H]}$ *and an unembedding matrix* U*, with at most* m *nonzero entries in each parameter matrix, defining a* $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ *such that it is a universal transformer for the class* $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$*, and*

$$R_Q^{l,h}, R_K^{l,h} \in \{0,1\}^{m \times d}, \quad R_V^{l,h} \in \{0,1\}^{m \times m}, \quad U \in \{0,1\}^{m \times d_{\mathrm{in}}},$$

if the embedding dimension satisfies $m \geq \overline{m}$*.*

Proof sketch. The proof proceeds by showing that the universal transformer can simultaneously match (i) the attention patterns and (ii) the value transformations of any transformer. The outputs of any target transformer T in the class $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$ and universal transformer can be written as

$$T(X^0) = \sum_{(h_1,\ldots,h_L) \in [H]^L} A^{L,h_L} \cdots A^{1,h_1} X^0 W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L}$$

$$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}(X^0, E) = \sum_{(h_1,\ldots,h_L) \in [H]^L} \tilde{A}^{L,h_L} \cdots \tilde{A}^{1,h_1} X^0 E R_V^{1,h_1} \cdots R_V^{L,h_L} U$$

First, we ensure that for every layer $l \in [L]$ and head $h \in [H]$, the attention matrices match: $\tilde{A}^{l,h} = A^{l,h}$. This guarantees that the universal transformer follows the same attention pattern over tokens as the target transformer. Second, conditioned on matching the attention matrices, we construct the value transformations so that $ER_V^{1,h_1} \cdots R_V^{L,h_L} U = W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L}$. Therefore, the overall input-output map is identical. This reduces the problem to implementing the correct linear transformations along each attention path.

Construction. We explicitly construct the parameters to realize this behavior. The key idea is to encode the weights of the target transformer into the embedding matrix E, and each parameter matrix is either routing or reading out from certain block of E. The sparsity follows from the fact that each matrix only interacts with a small number of coordinates.

We interpret the embedding space as an H-ary tree, indexed by $(h_1, \ldots, h_l) \in [H]^l$ for $l \leq L$, each corresponding to a head sequence. These blocks represent all possible computation paths through the transformer. For each layer l and head h, the matrix $R_V^{l,h}$ is a diagonal $0/1$ *mask*. It keeps the nodes of depth $\geq l$ whose l-th head index is h, and zeros everything else. So $R_V^{1,h_1} \cdots R_V^{k,h_k}$ keeps exactly the nodes of depth $\geq k$ that extend $p = (h_1, \ldots, h_k)$. $R_Q^{l,h}$ (resp. $R_K^{l,h}$) is the sum, over all nodes of depth $l-1$, of the selectors of their h-th query (resp. key) sub-block. U is the sum of all leaf selectors.

We further provide a sparse construction for looped (weight-tied) transformers, where the same parameters are reused across layers. This leads to a more succinct universal architectural description where the number of parameter matrices is $O(H)$ instead of $O(HL)$ in the standard setting.

Theorem 3.2 (Sparse looped universal transformer)**.** *There exist fixed parameters $\{R_Q^h, R_K^h, R_V^h\}_{h\in[H]}$ and an unembedding matrix U, with at most m nonzero entries in each parameter matrix, defining a* looped-$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ *such that it is a looped universal transformer for the class* $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$*, and*

$$R_Q^h, R_K^h \in \{0,1\}^{m\times d}, \quad R_V^h \in \{0,1\}^{m\times m}, \quad U \in \{0,1\}^{m\times d_{\mathrm{in}}},$$

when the embedding dimension satisfies $m \geq \overline{m}$.

Note that all the above constructions work even if there is attention masking, e.g., causal masking for decoder type of transformer, since we are matching all the attention matrices.

3.2 Random universal transformer

The sparse constructions above demonstrate that universality is achievable and establish a sufficient embedding dimension. However, they rely on carefully designed, highly structured parameters. This raises a natural question: *How special are universal transformers?* We now show that universality is in fact a generic property. In particular, a randomly initialized transformer is universal almost surely.

Theorem 3.3 (Random universal transformer)**.** *Let $\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}\}_{l\in[L],\, h\in[H]}$ and U be drawn independently from an absolutely continuous distribution over $\mathbb{R}^{m\times d}$, $\mathbb{R}^{m\times d}$, $\mathbb{R}^{m\times m}$, and $\mathbb{R}^{m\times d_{\mathrm{in}}}$, respectively (e.g., with i.i.d. Gaussian entries). Then, with probability 1, the resulting transformer $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ is a universal transformer for the class $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$, provided that $m \geq \overline{m}$.*

The universality condition can be reduced to a collection of algebraic non-degeneracy conditions on the parameters $\{R_Q, R_K, R_V, U\}$. These conditions fail only when certain matrices are rank-deficient or satisfy polynomial constraints. The theorem follows from the fact that such degeneracies form a measure-zero set. This also applies to the looped setting.

Theorem 3.4 (Random looped universal transformer)**.** *Let $\{R_Q^h, R_K^h, R_V^h\}_{h\in[H]}$ and U be drawn independently from an absolutely continuous distribution over $\mathbb{R}^{m\times d}$, $\mathbb{R}^{m\times d}$, $\mathbb{R}^{m\times m}$, and $\mathbb{R}^{m\times d_{\mathrm{in}}}$, respectively (e.g., with i.i.d. Gaussian entries). Then, with probability 1, the resulting looped transformer* looped-$\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$ *is a looped universal transformer for the class $\mathrm{TF}_{H,L,d_{\mathrm{in}},d}$, when the embedding dimension satisfies $m \geq \overline{m}$.*

These results show that the embedding dimension $m = \Theta(H^L \max\{d, d_{\mathrm{in}}\})$ is sufficient for universality, and that almost all transformers of this size are universal.

Finite precision. In the explicit sparse construction, all universal-transformer parameters are $\{0,1\}$-valued matrices, which are represented exactly in floating point. The embedding E stores the parameters that also appear in the computation path of the target transformer, so the construction introduces no additional ill-conditioned computation; its numerical error is essentially the same error incurred when evaluating the corresponding target transformer at the same precision. For the random construction, the potential source of numerical error arises when solving the system of linear equations. Its magnitude is controlled by the condition number of the random parameters. Thus, the simulation should remain stable for well-conditioned draws, although almost-sure invertibility alone does not provide a uniform finite-precision guarantee.

3.3 Beyond Attention-Only

The constructions in the previous sections are for attention-only transformers. Practical architectures interleave attention with residual connections, MLP blocks, and normalization. In this section we show that universality extends *exactly* to transformers with residual connections and gated MLPs (GLU variants, including SwiGLU, GeGLU, and ReGLU). We then explain which components break the argument.

MLP blocks. We can extend both the explicit sparse and random universality constructions to bias-free MLPs with any gated linear unit (GLU) activation σ of the form $MLP(X) = [\sigma(XW_g) \odot XW_u]W_d$, where $W_g, W_u \in \mathbb{R}^{d_{\text{in}} \times r}, W_d \in \mathbb{R}^{r \times d_{\text{in}}}$, r is the MLP width and σ is applied elementwise. The proof constructs auxiliary representations backward from the output so that the two linear branches (XW_g and XW_u) entering each gate are matched before applying σ. The main change is that one target neuron (neurons in W_d) needs to be simulated by a block of universal neurons. Because the universal transformer's output directions are fixed, and the target neuron $W_d[j,:] \in \mathbb{R}^{1 \times d_{\text{in}}}$, we use d_{in} universal neurons which form a set of basis to simulate a target neuron and encode the weights for each basis into the embedding matrix E and propagate through the linear branch (XW_u). If the target MLP width is r, and we need universal MLP to have width rd_{in}. A sufficient embedding dimension for E becomes $m = O\left((2Hr)^L(d_{\text{in}} + d)\right)$. See Appendix E for the details.

Residual connections. Adding residual connections, each layer update becomes $Y^l := X^{l-1} + \sum_{h=1}^{H} A^{l,h} X^{l-1} W_V^{l,h} W_O^{l,h}$, $X^l := Y^l + \left[\sigma(Y^l W_g^l) \odot (Y^l W_u^l)\right] W_d^l$. The proof extends directly by viewing the residual path as an additional fixed "head" carrying the identity transformation. The same path-unrolling argument then applies after replacing H by $H+1$ in the embedding-dimension bound. See Appendix E for details.

Layer normalization. Layernorm takes the input and normalize it by the mean and variance across the embedding dimension. For an input $X \in \mathbb{R}^{n \times d}$, define $\mu = \frac{1}{d} X \mathbf{1}_d \in \mathbb{R}^{n \times 1}$, $X_c = X - \mu \mathbf{1}_d^\top$. The row-wise variance is $v = \frac{1}{d}(X_c \odot X_c)\mathbf{1}_d \in \mathbb{R}^{n \times 1}$. Then $\text{LN}(X) = \left(\frac{X - \mu \mathbf{1}_d^\top}{\sqrt{v + \epsilon}\mathbf{1}_d^\top}\right) \odot \mathbf{1}_n \gamma^\top + \mathbf{1}_n \beta^\top$ where the division is elementwise. For universal transformers, embedding into a different dimension m changes the mean and variance across coordinates. Therefore, the linear representation invariant used in our proof is not preserved after normalization. Moreover, it's impossible for post-LayerNorm with arbitrary learned scale and bias. At $X = 0$, we have $XE = 0$ for every E, so the universal transformer produces one fixed output, whereas a target transformer can produce an arbitrary LayerNorm bias (β). Similarly, an arbitrary target scale (γ) can make its output arbitrarily large, while a fixed universal LayerNorm followed by a fixed unembedding has uniformly bounded output. Thus, extending our theorem to standard learned LayerNorm requires either restricting its affine parameters or allowing additional trainable parameters beyond the input embedding.

4 Optimality

In this section, we establish lower bounds on the embedding dimension m required for universality. Our upper bounds in Section 3 are achieved by explicitly matching the attention matrices of the target transformer, i.e., enforcing $\tilde{A}^{l,h} = A^{l,h}$ for all (l,h). We prove a lower bound on m for such "attention-matching" universal transformers. To do this, we prove a lower bound on m for a strictly stronger class of models that is additionally *given* the desired attention matrices as input. This, in turn, implies an $H^{\Omega(L)}$ lower bound on the embedding dimension for attention-matching approaches.

Shared-attention universal computation. We formalize this stronger model as follows. The shared-attention universal transformer, denoted shared-attn-UTF$^m_{H,L,d_{\text{in}},d}$, is parameterized only by value matrices $\{R_V^{l,h}\}_{l\in[L],h\in[H]}$. It takes as input a sequence $X^0 \in \mathbb{R}^{n\times d_{\text{in}}}$, an embedding matrix $E \in \mathbb{R}^{d_{\text{in}}\times m}$, an unembedding matrix $U \in \mathbb{R}^{m\times d_{\text{in}}}$, and attention matrices $\{A^{l,h}\}_{l\in[L],h\in[H]}$. The computation proceeds identically to UTF$^m_{H,L,d_{\text{in}},d}$, except that the attention matrices are externally provided and fixed, i.e., $\tilde{A}^{l,h}$ is replaced by $A^{l,h}$ for all (l,h).

Theorem 4.1 (Lower bound for universal transformers)**.** *Let $d_{\text{in}} = 1$. Consider any transformer class* TF$_{H,L,1,d}$*. If there exists a* shared-attn-UTF$^m_{H,L,d_{\text{in}},d}$ *(or* looped-shared-attn-UTF$^m_{H,L,d_{\text{in}},d}$*) such that for any $T \in$ TF$_{H,L,1,d}$ with attention matrices $\{A^{l,h}\}_{l\in[L],h\in[H]}$,*

$$\text{shared-attn-UTF}^m_{H,L,d_{\text{in}},d}(X, E, U, \{A^{l,h}\}_{l\in[L],h\in[H]}) = T(X)$$

for all $X \in \mathbb{R}^{n\times d_{\text{in}}}, n \in \mathbb{N}$, and $n \times n$ row-stochastic matrices $A^{l,h}$, then $m \geq H^{L/2}$.

Theorem 4.1 shows that the embedding dimension must grow exponentially in the depth L, even in the simplest setting $d_{\text{in}} = 1$. Notably, this lower bound holds in the shared-attention model, where the desired attention matrices are given as part of the input. Thus, the complexity arises from representing the exponentially-many computation paths induced by the multi-head attention structure.

In particular, across L layers and H heads, there are H^L possible sequences of attention heads, each corresponding to a distinct computation path. The universal transformer must be able to simultaneously represent and combine these paths. The lower bound shows that compressing this exponential-size family of behaviors into a low-dimensional embedding is impossible.

5 Experimental studies

In this section, we experimentally evaluate the universal transformers constructed in Section 3 on two canonical reasoning tasks—parenthesis balancing and k-hop induction heads—which are widely used as toy benchmarks for studying algorithmic reasoning in language models [27, 21]. These experiments complement our theoretical results on representational power by demonstrating empirically that standard gradient-based optimization can successfully learn the required embeddings for universal transformers to solve these tasks.

5.1 Tasks

Parenthesis Balancing (Dyck-1). In this task, the model is given a sequence of left- and right-parentheses and must output whether the parentheses in the sequence are properly balanced. Such sequences form the Dyck-1 language, a canonical example of a context-free language [4]. In our setup, the vocabulary size is 4: two parenthesis tokens and two labels. The maximum length for the input parentheses sequences is 60.

k-hop induction heads. This task generalizes the standard induction head mechanism studied in language models [9, 18]. In the standard induction heads, the model outputs the token that follows the most recent previous occurrence of the current token in the context. The k-hop variant extends this mechanism by recursively composing k such retrieval steps. For example, for an input $X = aabccabca$, the 2-hop induction head output for the last token a is c. Under a widely-believed complexity assumption, $\Theta(\log k)$ layers are necessary and sufficient to solve this task [21]. Therefore, we use this task to evaluate the performance of deeper universal transformers. For this task, the vocabulary size is 30, including the number of hops tokens, 4 character tokens and label tokens. The maximum sequence length is 100.

Architectures. We implement the universal transformer UTF$^m_{H,L,d_{\text{in}},d}$ using one-hot token representations, so that d_{in} corresponds to the vocabulary size. Following the construction in Theorem 3.1, for head dimension d, the embedding dimension is given by $m = \overline{m}$. We compare three models: (i) the constructed sparse universal transformer from Theorem 3.1, (ii) a random universal transformer with parameters initialized i.i.d. from Uniform$(-1/\sqrt{m}, 1/\sqrt{m})$ (the default PyTorch initialization), and (iii) a fully trained transformer. Following [29], for both the sparse and random universal transformers, we train only the embedding matrix E and the unembedding matrix U, while keeping all other parameters fixed. For random universal transformers, all the results are averaged across 5 seeds.

Task	Sparse	Random	Fully-trained
Parenthesis Balancing	100.0%	94.4%	100.0%
1-hop Induction Head	30.2%	29.4%	8.4%
2-hop Induction Head	33.2%	30.5%	51.8%
3-hop Induction Head	39.2%	32.6%	99.5%
4-hop Induction Head	39.4%	35.2%	95.6%

(a) Attention-only.

Task	Sparse	Random	Fully-trained
Parenthesis Balancing	100.0%	100.0%	93.0%
1-hop Induction Head	100.0%	96.1%	100.0%
2-hop Induction Head	100.0%	85.7%	100.0%
3-hop Induction Head	100.0%	92.6%	100.0%
4-hop Induction Head	100.0%	89.2%	100.0%

(b) Adding residual and layer normalization.

Table 1: Accuracy on parenthesis balancing and k-hop induction-head tasks. Parenthesis balancing uses 4-head, 2-layer transformers with head dimension 24 and embedding dimension 1024. The 1-, 2-, 3-, and 4-hop induction-head tasks use 2-head transformers with depths $2, 3, 4, 4$, head dimensions $28, 28, 30, 30$, and embedding dimensions $456, 1024, 2280, 2280$, respectively.

Model	Sparse	Random	Fully-trained
Attention-only	100%	94.4%	100%
+ residual	100%	95.4%	100%
+ residual and LN	100%	100%	93.0%
+ MLP	100%	100%	100%
+ MLP and LN	100%	100%	94.4%

(a) Parenthesis Balancing.

Model	Sparse	Random	Fully-trained
Attention-only	33.2%	30.5%	51.8%
+ residual	53.8%	54.3%	58.4%
+ residual and LN	100%	85.7%	100%
+ MLP	54.6%	54.1%	100%
+ MLP and LN	100%	66.2%	100%

(b) 2-hop Induction Heads.

Table 2: Ablation study on parenthesis balancing and 2-hop induction-head tasks. Accuracies are reported for 4-head, 2-layer transformers with head dimension 24 and embedding dimension 1024 on parenthesis balancing, and for 2-head, 3-layer transformers with head dimension 28 and embedding dimension 1024 on 2-hop induction heads.

5.2 Results

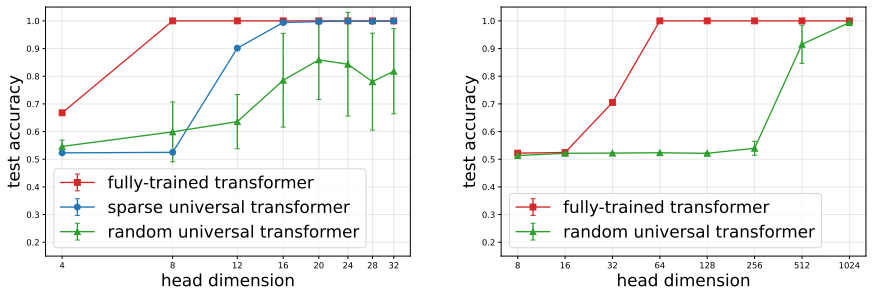
Table 1 summarizes the main results. We first consider the attention-only setting, which directly matches our theoretical model $\mathrm{UTF}^m_{H,L,d_{\mathrm{in}},d}$. On parenthesis balancing, the sparse universal transformer reaches 100.0% accuracy, matching the fully trained transformer and the random universal also reaches non-trivial accuracy 94.4%. This shows that the explicit construction from Theorem 3.1 is not merely a worst-case representation result: the embedding and unembedding matrices can be learned by gradient descent to solve a nontrivial algorithmic task.

The induction-head tasks are more challenging in the attention-only setting. Both sparse and random universal transformers obtain nontrivial accuracies (random guessing gives 20%) but far from optimal. This suggests that while the attention-only universal transformers have the representational capacity, optimization becomes increasingly difficult as the task requires more depth. Adding standard components that stabilize training can substantially improve performance. With residual connections and layer normalization, both sparse and random universal transformers reach much higher accuracy. In particular, the sparse universal transformer reaches 100.0% accuracy on all reported induction-head tasks, while the random universal transformer achieves between 85.7% and 96.1% accuracy.

5.2.1 Ablation studies

We further study how different components of standard transformers can affect the training optimization performance. Table 2 studies the effect of adding residual connections, layer normalization, and MLP layers. The MLP layers are randomly initialized using the PyTorch default initialization scheme. For the sparse and random universal transformers, the MLP layers are also kept fixed and untrained.

On parenthesis balancing, the sparse universal transformer is robust across all variants and achieves 100.0% accuracy in every setting. The random universal transformer also improves from 94.4% in the attention-only setting to 100.0% once residual connections and layer normalization or MLP layers are included. The 2-hop induction-head task reveals a sharper distinction between architectural components. Adding layer normalization together with residual connections can largely improve the performance, and this is also robust when adding frozen random MLP layers.



(a) Constructed parameterization $m = \overline{m}$ (b) Standard parametrization $m = Hd$

Figure 2: **Embedding dimension scaling for 2-hop induction heads.** Results are shown for 2-head, 3-layer transformers. *Left:* Performance under constructed parameterization from Theorem 3.1, where $m = \overline{m}$. *Right:* Performance under standard parameterization, where $m = Hd$. Error bars indicate ± 1 standard deviation across 5 random seeds.

Overall, the sparse construction gives a useful inductive bias for optimization: it consistently matches or exceeds the random universal transformer when only the embedding and unembedding are trained.

5.2.2 Embedding dimension scaling

We next study how performance empirically depends on the embedding dimension m. Our theory shows that the universal transformer exists when the embedding dimension grows with the number of attention paths, on the order of H^L up to factors depending on d and d_{in}, but this does not account for the optimization difficulty of finding the desired embedding. Therefore, we experimentally investigate how large m must be for standard gradient-based training to find a good solution. Figure 2 compares two parameterizations on the 2-hop induction-head task. In the constructed parameterization, we set $m = \overline{m}$ according to Theorem 3.1; in the standard parameterization, we instead use $m = Hd$, matching the usual transformer hidden dimension. This gives the empirical evidence that universal transformer can be found when the embedding dimension m is sufficiently large.

6 Conclusion and future work

In this paper, we study the universal transformer: a fixed-parameter transformer that can simulate any transformer in a prescribed class by changing only the input embedding. In this view, the embedding matrix acts as a description of the target model, while the internal attention and value parameters define a fixed computational template. We gave explicit sparse constructions of universal transformers, extended the construction to the looped weight-tied setting, and showed that universality is ubiquitous: randomly initialized transformers are universal almost surely.

As discussed in Section 3.3, extending these results to architectures with normalization or MLPs with non-gated activation is an important direction for future work. Our empirical results suggest that this extension is especially promising, since layernorm substantially improve performance in our experiments. We also note that our lower bound is proved only for a restricted matching-attention model, in which the universal transformer must match the target transformer's attention patterns. Establishing lower bounds for more general forms of simulation remains an important open problem.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Science Foundation under grant DMS-2502259, the Office of Naval Research under grant N00014-24-1-2700, a JP Morgan Faculty Award, and a Research Award from the Columbia Center of AI Technology in collaboration with Amazon. We are grateful to Ezra Edelman, for their helpful feedback on the construction of universal transformers, Surbhi Goel and Enric Boix-Adsera for pointing out the possibility to extend to MLP. We are also grateful to Sanjoy Dasgupta for telling us about [29].

References

[1] Vaggos Chatziafratis, Ioannis Panageas, Clayton Sanford, and Stelios Stavroulakis. On scrambling phenomena for randomly initialized recurrent networks. *Advances in Neural Information Processing Systems*, 35:18501–18513, 2022.

[2] Pin-Yu Chen. Model reprogramming: Resource-efficient cross-domain machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 22584–22591, 2024.

[3] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. In *Advances in Neural Information Processing Systems 33*, pages 15834–15846, 2020.

[4] Noam Chomsky and Marcel P. Schützenberger. The algebraic theory of context-free languages. *Studies in Logic and the Foundations of Mathematics*, 35:118–161, 1963.

[5] Stephen Chung and Hava Siegelmann. Turing completeness of bounded-precision recurrent neural networks. In *Advances in Neural Information Processing Systems 34*, pages 28431–28441, 2021.

[6] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

[7] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019.

[8] Yihe Dong, Lorenzo Noci, Mikhail Khodak, and Mufan Li. Is random attention sufficient for sequence modeling? disentangling trainable components in the transformer. *arXiv preprint arXiv:2506.01115*, 2025.

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

[10] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

[11] Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989.

[12] Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning*, 2023.

[13] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[14] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31*, 2018.

[15] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, GMD - German National Research Institute for Computer Science, 2001.

[16] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[17] Radford M Neal. Priors for infinite networks. In *Bayesian learning for neural networks*, pages 29–53. Springer, 1996.

[18] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[19] Hikari Otsuka, Daiki Chijiwa, Yasuyuki Okoshi, Daichi Fujiki, Susumu Takeuchi, and Masato Motomura. The strong lottery ticket hypothesis for multi-head attention mechanisms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 24639–24647, 2026.

[20] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20*, 2007.

[21] Clayton Sanford, Daniel Hsu, and Matus Telgarsky. Transformers, parallel computation, and logarithmic depth. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[22] Hava T Siegelmann and Eduardo D Sontag. On the computational power of neural nets. In *Fifth Annual ACM Conference on Computational Learning Theory*, pages 440–449, 1992.

[23] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

[24] Leslie G Valiant. Universal circuits (preliminary report). In *Proceedings of the eighth annual ACM symposium on Theory of computing*, pages 196–203, 1976.

[25] Colin Wei, Yining Chen, and Tengyu Ma. Statistically meaningful approximation: a case study on approximating turing machines with transformers. *Advances in Neural Information Processing Systems*, 35:12071–12083, 2022.

[26] Chao-Han Huck Yang, Yun-Yun Tsai, and Pin-Yu Chen. Voice2series: Reprogramming acoustic models for time series classification. In *International conference on machine learning*, pages 11808–11819. PMLR, 2021.

[27] Shunyu Yao, Binghui Peng, Christos Papadimitriou, and Karthik Narasimhan. Self-attention networks can process bounded hierarchical languages. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3770–3785, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.292. URL `https://aclanthology.org/2021.acl-long.292/`.

[28] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020.

[29] Ziqian Zhong and Jacob Andreas. Algorithmic capabilities of random transformers. In *Advances in Neural Information Processing Systems 37*, 2024.

A Proof of Theorem 3.1 (universal transformer)

Given an input X^0, let $Y := X^L$ denote the output of the target transformer, and $Z := \tilde{X}^L U$ denote the output of the universal transformer. We prove that for any target transformer, there exists a construction of E such that our constructed universal transformer outputs $Z = Y$ for any input X^0.

A.1 Warmup: single head per layer

In this section, we assume $H = 1$ and drop the h superscript in the notations.

Lemma A.1 (Unrolling). *Assume $H = 1$. For each $l \in [L]$,*

$$X^l = A^l \cdots A^1 X^0 W_V^1 W_O^1 \cdots W_V^l W_O^l$$
$$\tilde{X}^l = \tilde{A}^l \cdots \tilde{A}^1 X^0 E R_V^1 \cdots R_V^l.$$

Lemma A.2 (Equal attention). *Assume $H = 1$, and also that*

$$\begin{cases} W_V^1 W_O^1 \cdots W_V^{l-1} W_O^{l-1} W_Q^l = E R_V^1 \cdots R_V^{l-1} R_Q^l \\ W_V^1 W_O^1 \cdots W_V^{l-1} W_O^{l-1} W_K^l = E R_V^1 \cdots R_V^{l-1} R_K^l \end{cases} \quad \text{for each } l \in [L]. \tag{1}$$

Then for any $X^0 \in \mathbb{R}^{n \times d_{\text{in}}}$, we have $A^l = \tilde{A}^l$ for each $l \in [L]$.

Proof. The proof is by induction. The base case is $L = 1$. We have

$$X^0 W_Q^1 = X^0 E R_Q^1 = \tilde{X}^0 R_Q^1$$
$$X^0 W_K^1 = X^0 E R_K^1 = \tilde{X}^0 R_K^1$$

by (1) for $l = 1$ and the definition of $\tilde{X}^0$. Using these identities with the definitions of A^1 and $\tilde{A}^1$, we have

$$A^1 = \text{softmax}((X^0 W_Q^1)(X^0 W_K^1)^\intercal) = \text{softmax}((\tilde{X}^0 R_Q^1)(\tilde{X}^0 R_K^1)^\intercal) = \tilde{A}^1,$$

This proves the claim for $L = 1$.

For the inductive step, consider $L > 1$. We assume, as the inductive hypothesis, that for each $l \in [L-1]$, $A^l = \tilde{A}^l$. We need to show that $A^L = \tilde{A}^L$. We have

$$\begin{aligned} X^{L-1} W_Q^L &= A^{L-1} \cdots A^1 X^0 W_V^1 W_O^1 \cdots W_V^{L-1} W_O^{L-1} W_Q^L && \text{(by Lemma A.1)} \\ &= \tilde{A}^{L-1} \cdots \tilde{A}^1 X^0 W_V^1 W_O^1 \cdots W_V^{L-1} W_O^{L-1} W_Q^L && \text{(by inductive hypothesis)} \\ &= \tilde{A}^{L-1} \cdots \tilde{A}^1 X^0 E R_V^1 \cdots R_V^{L-1} R_Q^L && \text{(by (1) for } l = L) \\ &= \tilde{X}^{L-1} R_Q^L && \text{(by Lemma A.1),} \end{aligned}$$

and similarly,

$$X^{L-1} W_K^L = \tilde{X}^{L-1} R_K^L.$$

Therefore

$$A^L = \text{softmax}((X^{L-1} W_Q^L)(X^{L-1} W_K^L)^\intercal) = \text{softmax}((\tilde{X}^{L-1} R_Q^L)(\tilde{X}^{L-1} R_K^L)^\intercal) = \tilde{A}^L.$$

This completes the inductive step. □

Lemma A.3 (Equal input-output behavior). *Assume $H = 1$, and also that both (1) and*

$$W_V^1 W_O^1 \cdots W_V^L W_O^L = E R_V^1 \cdots R_V^L U \tag{2}$$

hold. Then for any $X^0 \in \mathbb{R}^{n \times d_{\text{in}}}$, we have $Y = Z$.

Proof. We have

$$\begin{aligned} Y = X^L &= A^L \cdots A^1 X^0 W_V^1 W_O^1 \cdots W_V^L W_O^L && \text{(by Lemma A.1)} \\ &= \tilde{A}^L \cdots \tilde{A}^1 X^0 W_V^1 W_O^1 \cdots W_V^L W_O^L && \text{(by Lemma A.2)} \\ &= \tilde{A}^L \cdots \tilde{A}^1 X^0 E R_V^1 \cdots R_V^L U && \text{(by (2))} \\ &= \tilde{X}^L U = Z && \text{(by Lemma A.1).} \end{aligned}$$

□

Lemma A.4 (Construction for single head)**.** *Assume $H = 1$ and $m \geq (L+1)\max\{2d, d_{\text{in}}\}$. Then there exists fixed $\{R_Q^l, R_K^l, R_V^l\}_{l\in[L]}$ and U such that for any transformer $T \in \text{TF}_{1,L,d_{\text{in}},d}$, there exists an E so that (1) and (2) hold.*

Proof. Let $b := \max\{2d, d_{\text{in}}\}$. Partition the first $(L+1)b$ embedding coordinates into consecutive blocks $B_1, \ldots, B_{L+1}$ of width b; any remaining coordinates are unused. Let $S_i \in \mathbb{R}^{m\times b}$ be the coordinate selector for B_i, so that ES_i is the submatrix of E stored in that block. Within a block, let $J_Q, J_K \in \mathbb{R}^{b\times d}$ select its first d and next d coordinates, respectively, and let $J_U \in \mathbb{R}^{b\times d_{\text{in}}}$ select its first d_{in} coordinates.

Define a common shift matrix

$$R := \sum_{i=1}^{L} S_{i+1}S_i^\intercal \in \mathbb{R}^{m\times m}.$$

Thus $RS_i = S_{i+1}$ for $i \leq L$, $RS_{L+1} = 0$, and R is zero on unused coordinates. Equivalently, right multiplication by R shifts the contents of B_{i+1} into B_i. For every layer $l \in [L]$, set

$$R_V^l := R, \qquad R_Q^l := S_1 J_Q, \qquad R_K^l := S_1 J_K, \qquad U := S_1 J_U.$$

In particular, these matrices are independent of the target transformer and are tied across layers. Since $R^t S_1 = S_{t+1}$ for $0 \leq t \leq L$, their products read the desired blocks of the original embedding.

Now fix an arbitrary target transformer $T \in \text{TF}_{1,L,d_{\text{in}},d}$ and write

$$W_{VO}^{1:l} := W_V^1 W_O^1 \cdots W_V^l W_O^l, \qquad W_{VO}^{1:0} := I_{d_{\text{in}}}.$$

For each $l \in [L]$, fill the first d columns of block B_l with $W_{VO}^{1:l-1}W_Q^l$, and the next d columns with $W_{VO}^{1:l-1}W_K^l$. Fill the first d_{in} columns of B_{L+1} with $W_{VO}^{1:L}$. Set all remaining entries of E to zero. These assignments do not overlap because $b \geq 2d$ and $b \geq d_{\text{in}}$.

For every $l \in [L]$, the selector identities give

$$ER_V^1 \cdots R_V^{l-1}R_Q^l = ER^{l-1}S_1 J_Q = ES_l J_Q = W_{VO}^{1:l-1}W_Q^l,$$
$$ER_V^1 \cdots R_V^{l-1}R_K^l = ER^{l-1}S_1 J_K = ES_l J_K = W_{VO}^{1:l-1}W_K^l.$$

Hence (1) holds. Similarly,

$$ER_V^1 \cdots R_V^L U = ER^L S_1 J_U = ES_{L+1}J_U = W_{VO}^{1:L},$$

proving (2). The construction works for every target transformer, including targets whose weights differ across layers. Each fixed matrix has entries in $\{0, 1\}$ and at most m nonzero entries. □

A.2 Multiple heads per layer

Lemma A.5 (Multi-headed Unrolling)**.** *For each $l \in [L]$,*

$$X^l = \sum_{(h_1,\ldots,h_l)\in[H]^l} A^{l,h_l}\cdots A^{1,h_1}X^0 W_V^{1,h_1}W_O^{1,h_1}\cdots W_V^{l,h_l}W_O^{l,h_l}$$
$$\tilde{X}^l = \sum_{(h_1,\ldots,h_l)\in[H]^l} \tilde{A}^{l,h_l}\cdots \tilde{A}^{1,h_1}X^0 E R_V^{1,h_1}\cdots R_V^{l,h_l}.$$

Proof. We prove the first identity; the second follows by the same argument with X^0 replaced by X^0E, $(W_V^{t,h_t}W_O^{t,h_t})$ replaced by R_V^{t,h_t}, and (A^{t,h_t}) replaced by $(\tilde{A}^{t,h_t})$. For $l = 1$, the update rule gives

$$X^1 = \sum_{h_1=1}^{H} A^{1,h_1}X^0 W_V^{1,h_1}W_O^{1,h_1},$$

which is exactly the stated formula. Assume the identity holds for $l-1 \geq 1$. Then

$$\begin{aligned} X^l &= \sum_{h_l=1}^{H} A^{l,h_l} X^{l-1} W_V^{l,h_l} W_O^{l,h_l} \\ &= \sum_{h_l=1}^{H} A^{l,h_l} \left(\sum_{(h_1,\ldots,h_{l-1}) \in [H]^{l-1}} A^{l-1,h_{l-1}} \cdots A^{1,h_1} X^0 \prod_{t=1}^{l-1} W_V^{t,h_t} W_O^{t,h_t} \right) W_V^{l,h_l} W_O^{l,h_l} \\ &= \sum_{(h_1,\ldots,h_l) \in [H]^l} A^{l,h_l} \cdots A^{1,h_1} X^0 \prod_{t=1}^{l} W_V^{t,h_t} W_O^{t,h_t}, \end{aligned}$$

which is the desired expansion. □

Lemma A.6 (Equal attention for multi-head)**.** *For each $l \in [L]$,*

$$\begin{cases} W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{l-1,h_{l-1}} W_O^{l-1,h_{l-1}} W_Q^{l,h_l} = E R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_Q^{l,h_l} \\ W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{l-1,h_{l-1}} W_O^{l-1,h_{l-1}} W_K^{l,h_l} = E R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_K^{l,h_l} \end{cases} \quad \text{for each } (h_1,\ldots,h_l) \in [H]^l. \tag{3}$$

Then for any $X^0 \in \mathbb{R}^{n \times d_{\text{in}}}$, we have $A^{l,h_l} = \tilde{A}^{l,h_l}$ for each $h_l \in [H], l \in [L]$.

Proof. Fix a layer $l \in [L]$ and a head $h_l \in [H]$. It suffices to show that

$$X^{l-1} W_Q^{l,h_l} = \tilde{X}^{l-1} R_Q^{l,h_l} \quad \text{and} \quad X^{l-1} W_K^{l,h_l} = \tilde{X}^{l-1} R_K^{l,h_l},$$

since then by the definitions of attention,

$$A^{l,h_l} = \operatorname{softmax}\big((X^{l-1} W_Q^{l,h_l})(X^{l-1} W_K^{l,h_l})^\top\big) = \operatorname{softmax}\big((\tilde{X}^{l-1} R_Q^{l,h_l})(\tilde{X}^{l-1} R_K^{l,h_l})^\top\big) = \tilde{A}^{l,h_l}.$$

By Lemma A.5, we can write

$$X^{l-1} W_Q^{l,h_l} = \sum_{(h_1,\ldots,h_{l-1}) \in [H]^{l-1}} A^{l-1,h_{l-1}} \cdots A^{1,h_1} X^0 \Big(W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{l-1,h_{l-1}} W_O^{l-1,h_{l-1}} W_Q^{l,h_l} \Big).$$

Using the assumption (3), each parenthesized product equals

$$E\, R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_Q^{l,h_l}.$$

Substituting and factoring out the rightmost R_Q^{l,h_l} gives

$$X^{l-1} W_Q^{l,h_l} = \sum_{(h_1,\ldots,h_{l-1}) \in [H]^{l-1}} A^{l-1,h_{l-1}} \cdots A^{1,h_1} X^0 E\, R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_Q^{l,h_l}.$$

If we assume (inductively in t) that for all $t \leq l-1$ and all heads h_t, we already have $A^{t,h_t} = \tilde{A}^{t,h_t}$, then the above becomes

$$\begin{aligned} X^{l-1} W_Q^{l,h_l} &= \sum_{(h_1,\ldots,h_{l-1}) \in [H]^{l-1}} \tilde{A}^{l-1,h_{l-1}} \cdots \tilde{A}^{1,h_1} X^0 E\, R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_Q^{l,h_l} \\ &= \tilde{X}^{l-1} R_Q^{l,h_l}, \end{aligned}$$

by the second identity in Lemma A.5. The W_K/R_K case is identical.

Finally, we prove the claim layer-by-layer. For $l=1$, the computation of A^{1,h_1} only depends on X^0 and the pair $(W_Q^{1,h_1}, W_K^{1,h_1})$, and the assumption (3) (with $l=1$) directly gives $X^0 W_Q^{1,h_1} = \tilde{X}^0 R_Q^{1,h_1}$ and $X^0 W_K^{1,h_1} = \tilde{X}^0 R_K^{1,h_1}$, hence $A^{1,h_1} = \tilde{A}^{1,h_1}$. Now assume for some $l \geq 2$ that $A^{t,h} = \tilde{A}^{t,h}$ holds for all $t \leq l-1$ and all heads $h \in [H]$. Repeating the argument above with this inductive hypothesis yields $X^{l-1} W_Q^{l,h_l} = \tilde{X}^{l-1} R_Q^{l,h_l}$ and $X^{l-1} W_K^{l,h_l} = \tilde{X}^{l-1} R_K^{l,h_l}$ for each h_l, and therefore $A^{l,h_l} = \tilde{A}^{l,h_l}$ for each h_l. This completes the induction. □

Lemma A.7 (Equal input-output behavior for multi-head)**.** *Assume (3) and*

$$W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L} = E R_V^{1,h_1} \cdots R_V^{L,h_L} U \quad \textit{for each } (h_1, \ldots, h_L) \in [H]^L \tag{4}$$

hold. Then for any $X^0 \in \mathbb{R}^{n \times d_{\text{in}}}$, we have $Y = Z$.

Proof. Fix $X^0 \in \mathbb{R}^{n \times d_{\text{in}}}$. By Lemma A.6, we have

$$A^{l,h} = \tilde{A}^{l,h} \qquad \text{for all } l \in [L],\ h \in [H].$$

Using the multi-headed unrolling Lemma A.5 at $l = L$, we can write

$$\begin{aligned} Y = X^L &= \sum_{(h_1,\ldots,h_L) \in [H]^L} A^{L,h_L} \cdots A^{1,h_1} X^0 \Big(W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L} \Big) \\ &= \sum_{(h_1,\ldots,h_L) \in [H]^L} \tilde{A}^{L,h_L} \cdots \tilde{A}^{1,h_1} X^0 \Big(W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L} \Big). \end{aligned}$$

By the assumption (4), each product in parentheses equals

$$E\, R_V^{1,h_1} \cdots R_V^{L,h_L} U.$$

Substituting this in, and then applying Lemma A.5 to recognize the resulting sum as $\tilde{X}^L$, yields

$$\begin{aligned} Y &= \sum_{(h_1,\ldots,h_L) \in [H]^L} \tilde{A}^{L,h_L} \cdots \tilde{A}^{1,h_1} X^0 E\, R_V^{1,h_1} \cdots R_V^{L,h_L} U \\ &= \tilde{X}^L U \\ &= Z, \end{aligned}$$

completing the proof. □

Theorem A.8 (Fixed sparse R_Q, R_K, R_V)**.** *Assume $H > 1$ and $m \geq N$, where*

$$N := \sum_{t=0}^{L-1} H^t \cdot (2Hd) + H^L d_{\text{in}} = O(H^L d + H^L d_{\text{in}}).$$

There exist fixed sparse matrices $R_Q^{l,h}, R_K^{l,h} \in \mathbb{R}^{m \times d}$, $R_V^{l,h} \in \mathbb{R}^{m \times m}$ for $l \in [L]$, $h \in [H]$, and $U \in \mathbb{R}^{m \times d_{\text{in}}}$ such that for every target transformer $T \in \text{TF}_{H,L,d_{\text{in}},d}$ there exists $E \in \mathbb{R}^{d_{\text{in}} \times m}$ satisfying (3) and (4).

Proof. **Construction idea.** We organize the active columns of E as an H-ary tree (Figure 3). The root stores the first-layer query and key matrices for all heads. Each internal node indexed by a head prefix stores the query and key matrices for the next layer, premultiplied by the value-output product along that prefix. Each leaf stores the complete value-output product along its path. The value maps act as diagonal masks: after choosing head h at layer l, they retain all descendant blocks whose l-th head is h and set the other blocks to zero. Thus they preserve the stored coefficients needed at subsequent layers. The query and key selectors read the appropriate subblocks at the current depth, and U reads the leaves.

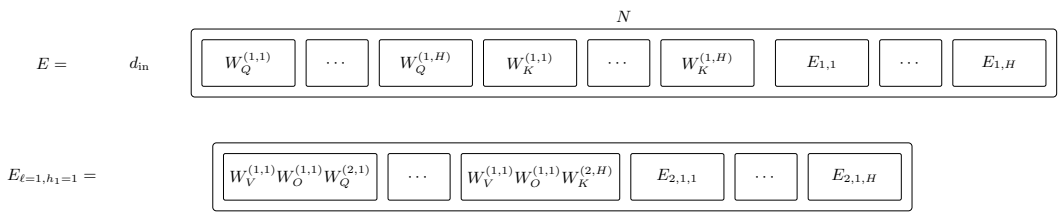


Figure 3: Active block structure of E and a schematic recursive expansion. Any additional embedding coordinates are set to zero.

For convenience, define

$$M^{l,h} := W_V^{l,h} W_O^{l,h} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}, \qquad M^{1:t,p} := \prod_{s=1}^{t} M^{s,h_s} \quad \text{for } p = (h_1, \ldots, h_t),$$

where products are ordered with increasing s from left to right and $M^{1:0,\emptyset} := I_{d_{\text{in}}}$.

Coordinate blocks. Partition the first N coordinates into disjoint blocks indexed by prefixes. For every $t \in \{0, \ldots, L-1\}$ and $p \in [H]^t$, allocate an internal block $B_{t,p}$ of width $2Hd$. For every $p \in [H]^L$, allocate a leaf block C_p of width d_{in}. Partition each internal block into subblocks

$$B_{t,p} = \Big(\bigsqcup_{h=1}^{H} B^Q_{t,p;h} \Big) \sqcup \Big(\bigsqcup_{h=1}^{H} B^K_{t,p;h} \Big), \qquad |B^Q_{t,p;h}| = |B^K_{t,p;h}| = d.$$

Any coordinates beyond N are unused. For a coordinate block D, write $S_D \in \mathbb{R}^{m \times |D|}$ for its coordinate selector, so that $ES_D = E_D$.

Define the embedding. For every internal block, set

$$E_{B^Q_{t,p;h}} = M^{1:t,p} W_Q^{t+1,h}, \qquad E_{B^K_{t,p;h}} = M^{1:t,p} W_K^{t+1,h}.$$

For every leaf, set

$$E_{C_p} = M^{1:L,p}.$$

Set all unused columns of E to zero. These assignments occupy disjoint blocks and specify an embedding for every target transformer.

Define the value maps. For each $l \in [L]$ and $h \in [H]$, let $R_V^{l,h}$ be diagonal, with

$$(R_V^{l,h})_{ii} = \begin{cases} 1, & i \in B_{t,p},\ t \geq l,\ p_l = h, \\ 1, & i \in C_p,\ p_l = h, \\ 0, & \text{otherwise.} \end{cases}$$

Here p_l denotes the l-th entry of the prefix p. Thus each map retains exactly the relevant descendant blocks, in their original coordinates. For a prefix $p = (h_1, \ldots, h_k)$, write

$$D_p := R_V^{1,h_1} \cdots R_V^{k,h_k}, \qquad D_\emptyset := I_m.$$

Since these matrices are diagonal, D_p retains an internal block $B_{t,q}$ precisely when $t \geq k$ and the first k entries of q equal p; it retains a leaf C_q precisely when its first k entries equal p. This also holds on all active blocks for $k = 0$.

Define the query, key, and output readouts. For $l \in [L]$ and $h \in [H]$, set

$$R_Q^{l,h} := \sum_{p \in [H]^{l-1}} S_{B^Q_{l-1,p;h}}, \qquad R_K^{l,h} := \sum_{p \in [H]^{l-1}} S_{B^K_{l-1,p;h}}, \qquad U := \sum_{p \in [H]^L} S_{C_p}.$$

The sums have dimensions $m \times d$, $m \times d$, and $m \times d_{\text{in}}$, respectively. These matrices are zero on unused coordinates.

Verify the multi-head QK equalities. Fix $l \in [L]$, $h_l \in [H]$, and $p = (h_1, \ldots, h_{l-1}) \in [H]^{l-1}$. Among the blocks at depth $l-1$, the mask D_p preserves precisely the block indexed by p. Consequently,

$$D_p R_Q^{l,h_l} = S_{B^Q_{l-1,p;h_l}}, \qquad D_p R_K^{l,h_l} = S_{B^K_{l-1,p;h_l}}.$$

This includes $l = 1$, when $D_p = I_m$ and the root is the only depth-zero block. Multiplication by E therefore gives

$$E R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_Q^{l,h_l} = E_{B^Q_{l-1,p;h_l}} = M^{1:l-1,p} W_Q^{l,h_l},$$
$$E R_V^{1,h_1} \cdots R_V^{l-1,h_{l-1}} R_K^{l,h_l} = E_{B^K_{l-1,p;h_l}} = M^{1:l-1,p} W_K^{l,h_l}.$$

Thus (3) holds, and Lemma A.6 implies $A^{l,h} = \tilde{A}^{l,h}$ for every layer and head.

Verify the multi-head OV equalities. For a full path $p = (h_1, \dots, h_L)$, the mask D_p preserves precisely its leaf block and annihilates every internal block and every other leaf. Hence $D_p U = S_{C_p}$, and

$$E R_V^{1,h_1} \cdots R_V^{L,h_L} U = E_{C_p} = M^{1:L,p} = W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L}.$$

This proves (4). By the equal input-output behavior lemma for multiple heads, the two transformers therefore have the same output for every X^0. All constructed matrices other than E depend only on $H, L, d, d_{\text{in}}, m$, and each has at most one nonzero entry per row. The same construction works for every $m \geq N$ by the zero padding already specified, completing the proof. □

B Proof of Theorem 3.3 (random universal transformer)

Theorem B.1 (Random parameters)**.** *Let $H, L \geq 1$ and assume*

$$m \geq N := 2Hd \sum_{t=0}^{L-1} H^t + d_{\text{in}} H^L.$$

Draw $\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}\}_{l\in[L],h\in[H]}$ and U independently from absolutely continuous distributions on their respective matrix spaces. Then, with probability 1, for every target transformer $T \in \text{TF}_{H,L,d_{\text{in}},d}$, there exists $E \in \mathbb{R}^{d_{\text{in}}\times m}$ such that (3) and (4) hold.

Proof. For a word $p = (h_1, \dots, h_t) \in [H]^t$, define

$$\mathcal{R}_p := R_V^{1,h_1} \cdots R_V^{t,h_t}, \qquad M_p := W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{t,h_t} W_O^{t,h_t},$$

with $\mathcal{R}_\phi := I_m$ and $M_\phi := I_{d_{\text{in}}}$ for the empty word ϕ. There are $H \sum_{t=0}^{L-1} H^t$ matrix constraints for each of Q and K, each contributing d columns, and H^L output constraints, each contributing d_{in} columns. Thus the total number of columns is N.

Stack all constraints into one linear system. Fix an ordering of the indices and form the horizontal concatenations

$$R_{\text{all}} := \Big[\{\mathcal{R}_p R_Q^{t+1,h}\}_{\substack{0\leq t<L,\ p\in[H]^t \\ h\in[H]}} \quad \{\mathcal{R}_p R_K^{t+1,h}\}_{\substack{0\leq t<L,\ p\in[H]^t \\ h\in[H]}} \quad \{\mathcal{R}_p U\}_{p\in[H]^L} \Big] \in \mathbb{R}^{m\times N},$$

$$W_{\text{all}} := \Big[\{M_p W_Q^{t+1,h}\}_{\substack{0\leq t<L,\ p\in[H]^t \\ h\in[H]}} \quad \{M_p W_K^{t+1,h}\}_{\substack{0\leq t<L,\ p\in[H]^t \\ h\in[H]}} \quad \{M_p\}_{p\in[H]^L} \Big] \in \mathbb{R}^{d_{\text{in}}\times N}.$$

Then (3) and (4) are equivalent to

$$E R_{\text{all}} = W_{\text{all}}. \tag{5}$$

If R_{all} has full column rank, its left inverse $R_{\text{all}}^\dagger = (R_{\text{all}}^\top R_{\text{all}})^{-1} R_{\text{all}}^\top$ satisfies $R_{\text{all}}^\dagger R_{\text{all}} = I_N$, and $E = W_{\text{all}} R_{\text{all}}^\dagger$ solves this system.

A nonvanishing determinant polynomial. Let ω collect all entries of the sampled query, key, value, and unembedding matrices. Every entry of R_{all} is a polynomial in ω. Let $\bar{R}_{\text{all}}$ be its submatrix consisting of the first N rows, and define

$$p(\omega) := \det(\bar{R}_{\text{all}}).$$

To show that this particular polynomial is nonzero, use the deterministic tied construction in the proof of Theorem C.1 below. That construction is independent of the present random-parameter argument: it gives, at dimension N, matrices for which the stacked columns are precisely the N distinct standard basis vectors. Assign these same tied matrices to every layer of the untied model. If $m > N$, embed the construction in the first N coordinates and set every other matrix entry to zero. At this assignment, $\bar{R}_{\text{all}}$ is a permutation matrix, so $p(\omega) = \pm 1$.

Consequently, p is not the zero polynomial. Its zero set has Lebesgue measure zero, and the joint law of ω is absolutely continuous. Therefore R_{all} has full column rank with probability 1. This event depends only on the sampled parameters, not on the target transformer. On this single event, the formula $E = W_{\text{all}} R_{\text{all}}^\dagger$ works for every target, proving the theorem.

For $H > 1$, $N = \frac{2H(H^L-1)}{H-1} d + H^L d_{\text{in}} = \overline{m}$. For $H = 1$, $N = 2Ld + d_{\text{in}} \leq (L+1)\max\{2d, d_{\text{in}}\} = \overline{m}$. Thus the embedding dimension assumed in Theorem 3.3 suffices in both cases. □

C Proof of Theorem 3.2 (looped universal transformer)

We give a tied construction whose query/key coordinates and output coordinates are disjoint. Only the simulating transformer is weight-tied; the target transformer may have different parameters at every layer.

Theorem C.1 (Designed R_Q, R_K, R_V for looped transformer)**.** *Let $H, L \geq 1$ and assume*

$$m \geq N := 2Hd \sum_{t=0}^{L-1} H^t + d_{\text{in}} H^L = \begin{cases} 2Ld + d_{\text{in}}, & H = 1, \\ \dfrac{2H(H^L - 1)}{H - 1} d + H^L d_{\text{in}}, & H > 1. \end{cases}$$

There exist fixed matrices

$$R_Q^h, R_K^h \in \{0,1\}^{m \times d}, \qquad R_V^h \in \{0,1\}^{m \times m} \quad (h \in [H]), \qquad U \in \{0,1\}^{m \times d_{\text{in}}},$$

each with at most m nonzero entries, such that for every target transformer $T \in \mathrm{TF}_{H,L,d_{\text{in}},d}$ there exists $E \in \mathbb{R}^{d_{\text{in}} \times m}$ satisfying (3) and (4) with $R_Q^{l,h} = R_Q^h$, $R_K^{l,h} = R_K^h$, and $R_V^{l,h} = R_V^h$. Consequently, the resulting looped transformer simulates T exactly for every input.

Proof. It suffices to construct the matrices for $m = N$. Larger dimensions are handled by placing this construction in the first N coordinates, setting all remaining entries of the fixed matrices to zero, and appending zero columns to E.

Separate query/key and output coordinates. Index the standard basis of $\mathbb{R}^N$ by the following two disjoint families:

$$e^{\sigma}_{p,h,j}, \qquad \sigma \in \{Q, K\}, \quad p \in \bigcup_{t=0}^{L-1} [H]^t, \quad h \in [H], \quad j \in [d],$$

and

$$f_{s,a}, \qquad s \in [H]^L, \quad a \in [d_{\text{in}}].$$

The first family has $2Hd \sum_{t=0}^{L-1} H^t$ vectors, and the second has $d_{\text{in}} H^L$ vectors, giving exactly N coordinates. The empty word is denoted by ϕ.

Define the tied matrices. For each $g \in [H]$, define R_V^g on the query/key basis vectors by

$$R_V^g e^{\sigma}_{p,h,j} := \begin{cases} e^{\sigma}_{(g,p),h,j}, & |p| < L - 1, \\ 0, & |p| = L - 1. \end{cases}$$

Thus the operator prepends a head index until the maximum query/key depth is reached. On the output basis vectors, define

$$R_V^g f_{(s_1,\ldots,s_L),a} := f_{(g,s_1,\ldots,s_{L-1}),a}.$$

This operator prepends g and discards the last head index. For $L = 1$, the expression on the right is $f_{(g),a}$. Define the columns of the query, key, and output matrices by

$$(R_Q^h)_{:,j} := e^Q_{\phi,h,j}, \qquad (R_K^h)_{:,j} := e^K_{\phi,h,j}, \qquad U_{:,a} := f_{(1,\ldots,1),a}.$$

These matrices are fixed independently of the target. Every column of each R_V^g is either zero or a standard basis vector, so R_V^g has at most N nonzero entries. Each R_Q^h and R_K^h has d nonzero entries, and U has d_{in} nonzero entries. All entries are in $\{0, 1\}$.

For $p = (h_1, \ldots, h_t)$, write

$$R_p := R_V^{h_1} \cdots R_V^{h_t}, \qquad R_\phi := I_N.$$

Because matrix products act on column vectors from right to left, prepending indices gives

$$(R_p R_Q^h)_{:,j} = e^Q_{p,h,j}, \qquad (R_p R_K^h)_{:,j} = e^K_{p,h,j} \qquad (0 \leq |p| \leq L - 1).$$

After exactly L output shifts, all entries of the initial word $(1, \ldots, 1)$ have been discarded, and therefore

$$(R_s U)_{:,a} = f_{s,a} \qquad (s \in [H]^L).$$

In particular, the stacked columns $\{R_p R_Q^h\}$, $\{R_p R_K^h\}$, and $\{R_s U\}$ are exactly the N distinct standard basis vectors. This fact will also supply the full-rank witness for the random-parameter results.

Encode the arbitrary target in E. For $p = (h_1, \ldots, h_t)$, define the layer-indexed target product

$$M_p := W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{t,h_t} W_O^{t,h_t}, \qquad M_\phi := I_{d_{\text{in}}}.$$

Let $\varepsilon_j \in \mathbb{R}^d$ and $\eta_a \in \mathbb{R}^{d_{\text{in}}}$ denote standard basis vectors. Specify E on its coordinate basis by

$$\begin{aligned} Ee^Q_{p,h,j} &:= M_p W_Q^{t+1,h} \varepsilon_j, && |p| = t < L, \\ Ee^K_{p,h,j} &:= M_p W_K^{t+1,h} \varepsilon_j, && |p| = t < L, \\ Ef_{s,a} &:= M_s \eta_a, && |s| = L. \end{aligned}$$

Since all these coordinates are disjoint, the prescriptions define one matrix $E \in \mathbb{R}^{d_{\text{in}} \times N}$ without conflicting assignments.

Verify the matching conditions. The basis identities above imply

$$ER_p R_Q^h = M_p W_Q^{t+1,h}, \qquad ER_p R_K^h = M_p W_K^{t+1,h} \qquad (|p| = t < L),$$

and

$$ER_s U = M_s \qquad (|s| = L).$$

These are precisely (3) and (4) for the tied simulating parameters and the layer-dependent target parameters. The equal-attention and equal-input-output lemmas therefore give $\tilde{A}^{l,h} = A^{l,h}$ for every layer and head, and $\tilde{X}^L U = X^L$ for every input.

The construction also applies when $H = 1$: the query/key coordinates form two families of chains, and each output coordinate is fixed by R_V^1. Its dimension is $2Ld + d_{\text{in}}$. For $H \geq 2$, the displayed dimension is $O(H^L(d + d_{\text{in}}))$. □

D Proof of Theorem 3.4 (random looped universal transformer)

Theorem D.1 (Random looped parameters)**.** *Let $H, L \geq 1$ and assume*

$$m \geq N := 2Hd \sum_{t=0}^{L-1} H^t + d_{\text{in}} H^L.$$

Let $\{R_Q^h, R_K^h, R_V^h\}_{h \in [H]}$ and U be drawn independently from absolutely continuous distributions on their respective matrix spaces, where $R_Q^h, R_K^h \in \mathbb{R}^{m \times d}$, $R_V^h \in \mathbb{R}^{m \times m}$, and $U \in \mathbb{R}^{m \times d_{\text{in}}}$. Then, with probability 1, for every target transformer $T \in \text{TF}_{H,L,d_{\text{in}},d}$ there exists $E \in \mathbb{R}^{d_{\text{in}} \times m}$ such that

$$M_p W_Q^{t+1,h} = ER_p R_Q^h, \qquad p \in [H]^t,\ 0 \leq t < L,\ h \in [H], \tag{6}$$

$$M_p W_K^{t+1,h} = ER_p R_K^h, \qquad p \in [H]^t,\ 0 \leq t < L,\ h \in [H], \tag{7}$$

$$M_s = ER_s U, \qquad s \in [H]^L. \tag{8}$$

Here, for $p = (h_1, \ldots, h_t)$,

$$R_p := R_V^{h_1} \cdots R_V^{h_t}, \qquad M_p := W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{t,h_t} W_O^{t,h_t},$$

with $R_\phi = I_m$ and $M_\phi = I_{d_{\text{in}}}$. Consequently, the sampled looped transformer simulates every such target through a suitable embedding E.

Proof. **Stack all constraints into one linear system.** Using the same ordering for both matrices, define

$$R_{\text{all}} := \Big[\{R_p R_Q^h\}_{\substack{0 \leq t < L,\ p \in [H]^t \\ h \in [H]}} \quad \{R_p R_K^h\}_{\substack{0 \leq t < L,\ p \in [H]^t \\ h \in [H]}} \quad \{R_s U\}_{s \in [H]^L} \Big] \in \mathbb{R}^{m \times N},$$

$$W_{\text{all}} := \Big[\{M_p W_Q^{t+1,h}\}_{\substack{0 \leq t < L,\ p \in [H]^t \\ h \in [H]}} \quad \{M_p W_K^{t+1,h}\}_{\substack{0 \leq t < L,\ p \in [H]^t \\ h \in [H]}} \quad \{M_s\}_{s \in [H]^L} \Big] \in \mathbb{R}^{d_{\text{in}} \times N}.$$

The conditions in the theorem are equivalent to

$$ER_{\text{all}} = W_{\text{all}}. \tag{9}$$

There are $H\sum_{t=0}^{L-1} H^t$ blocks of width d in each query/key group and H^L blocks of width d_{in} in the output group, giving the stated column count N.

A nonzero polynomial witness. Let x consist of all entries of the tied matrices $\{R_Q^h, R_K^h, R_V^h\}_{h\in[H]}$ and the sampled matrix U. Each entry of R_{all} is a polynomial in x, including when a tied matrix occurs repeatedly in a product. Define

$$p(x) := \sum_{\substack{I\subseteq[m]\\|I|=N}} \det\big(R_{\text{all}}[I,:]\big)^2.$$

Then $p(x) = 0$ if and only if R_{all} has rank smaller than N. The construction in Theorem C.1, embedded in the first N coordinates if necessary, makes these stacked columns distinct standard basis vectors. Hence one $N \times N$ minor has determinant ± 1, and p is not the zero polynomial.

The zero set of p has Lebesgue measure zero. Because the joint distribution of all sampled parameters, including U, is absolutely continuous, we have

$$\Pr\big(\operatorname{rank}(R_{\text{all}}) = N\big) = 1.$$

On this event, keep the sampled U fixed and set

$$E := W_{\text{all}} R_{\text{all}}^{\dagger}, \qquad R_{\text{all}}^{\dagger} := (R_{\text{all}}^{\top} R_{\text{all}})^{-1} R_{\text{all}}^{\top}.$$

Since $R_{\text{all}}^{\dagger} R_{\text{all}} = I_N$, this solves (9). The full-rank event is independent of the target, so the same sampled backbone and unembedding work for every target, with only E changing. The matching conditions then imply identical attention matrices and input-output behavior, as required. □

E Extension to SwiGLU feed-forward blocks

We extend the sparse and random constructions to bias-free attention layers followed by SwiGLU feed-forward blocks. The proof uses linear probes of each hidden representation and places target-dependent coefficients in the linear branch of the gate. Throughout this section, r is the target feed-forward hidden width and σ is a fixed entrywise activation. SwiGLU corresponds to $\sigma(t) = t/(1+\exp(-t))$; the argument does not require any additional property of σ.

Architecture. Write $M^{l,h} := W_V^{l,h} W_O^{l,h} \in \mathbb{R}^{d_{\text{in}}\times d_{\text{in}}}$. Starting from $X^0 \in \mathbb{R}^{n\times d_{\text{in}}}$, the target layer computes

$$\begin{aligned} A^{l,h} &:= \operatorname{softmax}\Big((X^{l-1}W_Q^{l,h})(X^{l-1}W_K^{l,h})^{\intercal}\Big), \\ S^l &:= \sum_{h=1}^{H} A^{l,h} X^{l-1} M^{l,h}, \\ X^l &:= \big[\sigma(S^l W_g^l) \odot (S^l W_u^l)\big]\, W_d^l, \end{aligned} \tag{10}$$

where $\odot$ denotes entrywise multiplication, $W_g^l, W_u^l \in \mathbb{R}^{d_{\text{in}}\times r}$ and $W_d^l \in \mathbb{R}^{r\times d_{\text{in}}}$. The query and key matrices have the same dimensions as in the attention-only model. For a fixed activation σ, denote this target class by $\mathrm{TF}^{\sigma}_{H,L,d_{\text{in}},d,r}$.

The universal model has embedding dimension m and layer-dependent feed-forward hidden widths q_l. Its fixed parameters are

$$\begin{aligned} R_Q^{l,h}, R_K^{l,h} &\in \mathbb{R}^{m\times d}, & R_V^{l,h} &\in \mathbb{R}^{m\times m}, \\ R_g^l, R_u^l &\in \mathbb{R}^{m\times q_l}, & R_d^l &\in \mathbb{R}^{q_l\times m}, \qquad U \in \mathbb{R}^{m\times d_{\text{in}}}. \end{aligned}$$

Given an embedding $E \in \mathbb{R}^{d_{\text{in}} \times m}$, it starts from $\tilde{X}^0 = X^0 E$ and computes

$$\begin{aligned}
\tilde{A}^{l,h} &:= \operatorname{softmax}\Big((\tilde{X}^{l-1} R_Q^{l,h})(\tilde{X}^{l-1} R_K^{l,h})^\top \Big), \\
\tilde{S}^l &:= \sum_{h=1}^{H} \tilde{A}^{l,h} \tilde{X}^{l-1} R_V^{l,h}, \\
\tilde{X}^l &:= \Big[\sigma(\tilde{S}^l R_g^l) \odot (\tilde{S}^l R_u^l) \Big] R_d^l.
\end{aligned} \tag{11}$$

Its output is $\tilde{X}^L U$. Only E depends on the target transformer.

Backward widths and probes. For positive integers $H, L, d_{\text{in}}, d, r$, define

$$s_L := d_{\text{in}}, \qquad q_l := r s_l, \qquad s_{l-1} := 2Hd + 2Hq_l \quad (l = L, \dots, 1). \tag{12}$$

With $a := 2Hr \geq 2$, this gives

$$s_0 = a^L d_{\text{in}} + 2Hd \frac{a^L - 1}{a - 1} = O\big((2Hr)^L (d_{\text{in}} + d)\big). \tag{13}$$

For each layer, partition $[q_l]$ into the consecutive groups

$$I_{l,j} := \{(j-1)s_l + 1, \dots, j s_l\}, \qquad j \in [r].$$

In the following concatenations, braces indexed by h mean horizontal concatenation in increasing head order, preserving the displayed order within each head.

Lemma E.1 (Backward linear-probe simulation)**.** *For fixed universal parameters, define $P_l \in \mathbb{R}^{m \times s_l}$ by $P_L = U$ and*

$$P_{l-1} := \Big[\{R_Q^{l,h}, R_K^{l,h}\}_{h \in [H]},\ \{R_V^{l,h} R_g^l, R_V^{l,h} R_u^l\}_{h \in [H]} \Big]. \tag{14}$$

Suppose that P_0 has full column rank and that every matrix

$$B_{l,j} := R_d^l[I_{l,j}, :] P_l \in \mathbb{R}^{s_l \times s_l} \tag{15}$$

is invertible. Then, for every target transformer $T \in \mathrm{TF}^{\sigma}_{H,L,d_{\text{in}},d,r}$, there exists $E \in \mathbb{R}^{d_{\text{in}} \times m}$ such that $\tilde{X}^L U = T(X^0)$ for every context length $n \in \mathbb{N}$ and every input $X^0 \in \mathbb{R}^{n \times d_{\text{in}}}$.

Proof. Construct target probes $C_l \in \mathbb{R}^{d_{\text{in}} \times s_l}$ backward, starting with $C_L = I_{d_{\text{in}}}$. Given C_l, define for each $j \in [r]$

$$\gamma_{l,j} := W_d^l[j, :] C_l, \qquad \alpha_{l,j} := \gamma_{l,j} B_{l,j}^{-1} \quad \text{in } \mathbb{R}^{1 \times s_l}. \tag{16}$$

For $i = (j-1)s_l + b \in I_{l,j}$, set

$$\widehat{W}_g^l[:, i] := W_g^l[:, j], \qquad \widehat{W}_u^l[:, i] := \alpha_{l,j,b} W_u^l[:, j]. \tag{17}$$

The resulting matrices $\widehat{W}_g^l, \widehat{W}_u^l$ have dimensions $d_{\text{in}} \times q_l$. Define the preceding target probe in the same column order as (14):

$$C_{l-1} := \Big[\{W_Q^{l,h}, W_K^{l,h}\}_{h \in [H]},\ \{M^{l,h} \widehat{W}_g^l, M^{l,h} \widehat{W}_u^l\}_{h \in [H]} \Big]. \tag{18}$$

Because P_0 has full column rank, choose a left inverse $P_0^\dagger$ and set $E = C_0 P_0^\dagger$. Thus $E P_0 = C_0$.

We prove by induction that

$$\tilde{X}^l P_l = X^l C_l \qquad (l = 0, \dots, L). \tag{19}$$

The base case follows from $\tilde{X}^0 = X^0 E$. Assume the identity at layer $l - 1$. The query and key columns in the probes give

$$\tilde{X}^{l-1} R_Q^{l,h} = X^{l-1} W_Q^{l,h}, \qquad \tilde{X}^{l-1} R_K^{l,h} = X^{l-1} W_K^{l,h},$$

so $\tilde{A}^{l,h} = A^{l,h}$ for every head. The remaining columns give, for $i = (j-1)s_l + b$,

$$\tilde{S}^l R_g^l[:, i] = \sum_{h=1}^{H} A^{l,h} X^{l-1} M^{l,h} W_g^l[:, j] = S^l W_g^l[:, j],$$

$$\tilde{S}^l R_u^l[:, i] = \alpha_{l,j,b} \sum_{h=1}^{H} A^{l,h} X^{l-1} M^{l,h} W_u^l[:, j] = \alpha_{l,j,b} S^l W_u^l[:, j].$$

Let $z_{l,j} := \sigma(S^l W_g^l[:, j]) \odot (S^l W_u^l[:, j]) \in \mathbb{R}^{n \times 1}$. The universal hidden unit i equals $\alpha_{l,j,b} z_{l,j}$, and consequently

$$\begin{aligned}
\tilde{X}^l P_l &= \sum_{j=1}^{r} z_{l,j} \alpha_{l,j} R_d^l[I_{l,j}, :] P_l \\
&= \sum_{j=1}^{r} z_{l,j} \alpha_{l,j} B_{l,j} = \sum_{j=1}^{r} z_{l,j} \gamma_{l,j} \\
&= \left[\sigma(S^l W_g^l) \odot (S^l W_u^l)\right] W_d^l C_l = X^l C_l.
\end{aligned}$$

This completes the induction. Since $P_L = U$ and $C_L = I_{d_{\text{in}}}$, its final identity is $\tilde{X}^L U = X^L$. The universal probes depend only on the fixed universal parameters. The target probes and the embedding may also depend on the target parameters, but none of these objects depends on n or X^0. □

Theorem E.2 (Fixed sparse parameters with SwiGLU blocks)**.** *Fix positive integers $H, L, d_{\text{in}}, d, r$ and an entrywise activation σ. Let s_l, q_l be defined by* (12)*, and assume $m \geq s_0$. There exist fixed parameters*

$$\begin{aligned}
R_Q^{l,h}, R_K^{l,h} &\in \{0,1\}^{m \times d}, & R_V^{l,h} &\in \{0,1\}^{m \times m}, \\
R_g^l, R_u^l &\in \{0,1\}^{m \times q_l}, & R_d^l &\in \{0,1\}^{q_l \times m},
\end{aligned}$$

for $l \in [L]$, $h \in [H]$, and an unembedding matrix $U \in \{0,1\}^{m \times d_{\text{in}}}$, each with at most m nonzero entries, such that for every target transformer $T \in \mathrm{TF}^{\sigma}_{H,L,d_{\text{in}},d,r}$ there exists $E \in \mathbb{R}^{d_{\text{in}} \times m}$ for which the universal computation in (11) *satisfies*

$$\tilde{X}^L U = T(X^0) \qquad \textit{for all } n \in \mathbb{N},\ X^0 \in \mathbb{R}^{n \times d_{\text{in}}}.$$

Proof. Let $e_1, \ldots, e_m$ be the standard basis of $\mathbb{R}^m$, and write $J_t := [e_1, \ldots, e_t]$. We construct parameters for which $P_l = J_{s_l}$ at every layer. Set $U = J_{d_{\text{in}}}$. At layer l, abbreviate $q = q_l$ and define, for $k \in [d]$,

$$R_Q^{l,h}[:, k] = e_{2(h-1)d+k}, \qquad R_K^{l,h}[:, k] = e_{(2h-1)d+k}.$$

Choose the two gate-input selectors

$$R_g^l = [e_1, \ldots, e_q], \qquad R_u^l = [e_{q+1}, \ldots, e_{2q}].$$

For $h \in [H]$ and $i \in [q]$, put

$$b_g(h, i) := 2Hd + 2(h-1)q + i, \qquad b_u(h, i) := 2Hd + (2h-1)q + i,$$

and define the value-routing matrices by

$$R_V^{l,h} := \sum_{i=1}^{q} \left(e_{b_g(h,i)} e_i^{\top} + e_{b_u(h,i)} e_{q+i}^{\top} \right). \tag{20}$$

These assignments make the concatenation in (14) exactly $J_{s_{l-1}}$. Finally, for $j \in [r]$ and $b \in [s_l]$, set

$$R_d^l[(j-1)s_l + b, :] = e_b^{\top}.$$

Then $R_d^l[I_{l,j}, :] P_l = I_{s_l}$ for every l, j, while $P_0 = J_{s_0}$ has full column rank.

All coordinates used are at most $s_{l-1} \leq s_0 \leq m$. Each query or key matrix has d nonzeros; each gate, up, or down matrix has q_l; each value matrix has $2q_l$; and U has d_{in}. These counts are at most m, since $s_{l-1} = 2Hd + 2Hq_l \geq 2q_l$. All entries are binary, and Lemma E.1 proves the claim. □

Theorem E.3 (Random parameters with SwiGLU blocks)**.** *Fix positive integers $H, L, d_{\text{in}}, d, r$ and an entrywise activation σ. Let s_l, q_l be defined by* (12)*, and assume $m \geq s_0$. Draw the parameters*

$$R_Q^{l,h}, R_K^{l,h} \in \mathbb{R}^{m\times d}, \qquad R_V^{l,h} \in \mathbb{R}^{m\times m},$$
$$R_g^l, R_u^l \in \mathbb{R}^{m\times q_l}, \qquad R_d^l \in \mathbb{R}^{q_l\times m},$$

for $l \in [L]$, $h \in [H]$, and $U \in \mathbb{R}^{m\times d_{\text{in}}}$ jointly from an absolutely continuous distribution on the product of their matrix spaces. Then, with probability 1*, for every target transformer $T \in \mathrm{TF}^{\sigma}_{H,L,d_{\text{in}},d,r}$ there exists $E \in \mathbb{R}^{d_{\text{in}}\times m}$ for which the universal computation in* (11) *satisfies*

$$\tilde{X}^L U = T(X^0) \qquad \text{for all } n \in \mathbb{N},\ X^0 \in \mathbb{R}^{n\times d_{\text{in}}}.$$

Proof. Construct the probes by (14), starting from $P_L = U$. Fix in advance the square minor formed by the first s_0 rows of P_0, and denote it by $\overline{P}_0 := P_0[[s_0], :]$. As a function of all universal parameters, consider

$$\Delta := \det(\overline{P}_0) \prod_{l=1}^{L} \prod_{j=1}^{r} \det\big(R_d^l[I_{l,j}, :]P_l\big)\,. \tag{21}$$

Every probe entry is a polynomial in the universal parameters, so Δ is a polynomial. At the explicit binary assignment in the proof of Theorem E.2, $\overline{P}_0 = I_{s_0}$ and every local matrix is I_{s_l}. Hence $\Delta = 1$ at that assignment, proving that the polynomial is not identically zero.

The zero set of a nonzero real polynomial has Lebesgue measure zero. Joint absolute continuity therefore gives $\Delta \neq 0$ with probability one. On this event, P_0 has full column rank and every local matrix in (15) is invertible. This event depends only on the fixed universal parameters. Thus Lemma E.1 applies simultaneously to every target transformer, giving the stated uniform almost-sure conclusion. □

Residual blocks. The same approach also accommodates a residual addition around each attention and feed-forward sublayer. Retain the attention definitions above, and replace the target updates by

$$Y^l := X^{l-1} + \sum_{h=1}^{H} A^{l,h} X^{l-1} M^{l,h},$$
$$X^l := Y^l + \left[\sigma(Y^l W_g^l) \odot (Y^l W_u^l)\right] W_d^l. \tag{22}$$

The universal model uses the corresponding updates

$$\tilde{Y}^l := \tilde{X}^{l-1} + \sum_{h=1}^{H} \tilde{A}^{l,h} \tilde{X}^{l-1} R_V^{l,h},$$
$$\tilde{X}^l := \tilde{Y}^l + \left[\sigma(\tilde{Y}^l R_g^l) \odot (\tilde{Y}^l R_u^l)\right] R_d^l. \tag{23}$$

Denote the target class in (22) by $\mathrm{TF}^{\sigma,\text{res}}_{H,L,d_{\text{in}},d,r}$.

Theorem E.4 (Residual SwiGLU blocks)**.** *Fix positive integers $H, L, d_{\text{in}}, d, r$ and an entrywise activation σ. Define the backward widths by*

$$s_L := d_{\text{in}}, \qquad q_l := r s_l, \qquad s_{l-1} := 2Hd + (H+1)s_l + 2(H+1)q_l, \tag{24}$$

for $l = L, \ldots, 1$, and assume $m \geq s_0$. There exist fixed parameters

$$\{R_Q^{l,h}, R_K^{l,h}, R_V^{l,h}\}_{l\in[L],h\in[H]}, \qquad \{R_g^l, R_u^l, R_d^l\}_{l\in[L]}, \qquad U,$$

with the matrix dimensions in Theorem E.2 using these widths q_l, entries in $\{0,1\}$, and at most m nonzero entries in each matrix, such that for every target transformer $T \in \mathrm{TF}^{\sigma,\text{res}}_{H,L,d_{\text{in}},d,r}$ there exists $E \in \mathbb{R}^{d_{\text{in}}\times m}$ for which the residual universal computation in (23) *satisfies*

$$\tilde{X}^L U = T(X^0) \qquad \text{for all } n \in \mathbb{N},\ X^0 \in \mathbb{R}^{n\times d_{\text{in}}}.$$

Moreover, if all these parameters are drawn jointly from an absolutely continuous distribution on the product of their real matrix spaces, then, with probability 1*, the same simulation property holds: for every such T there exists an embedding E that gives the displayed equality for all context lengths and inputs.*

Proof. We first establish the sufficient probe conditions, and then construct a simultaneous witness for them. Set $P_L = U$ and form each P_{l-1} by horizontally concatenating the following three blocks in order:

$$\begin{aligned}&\left[\{R_Q^{l,h}, R_K^{l,h}\}_{h\in[H]}\right],\\ &\left[P_l,\ \{R_V^{l,h}P_l\}_{h\in[H]}\right],\\ &\left[R_g^l, R_u^l,\ \{R_V^{l,h}R_g^l, R_V^{l,h}R_u^l\}_{h\in[H]}\right].\end{aligned} \tag{25}$$

The total number of columns is s_{l-1} in (24). Suppose P_0 has full column rank and all $B_{l,j} = R_d^l[I_{l,j},:]P_l$ are invertible. Starting with $C_L = I_{d_{\text{in}}}$, define $\gamma_{l,j}, \alpha_{l,j}, \widehat{W}_g^l, \widehat{W}_u^l$ by (16)–(17). Define C_{l-1} by the corresponding concatenation

$$\begin{aligned}&\left[\{W_Q^{l,h}, W_K^{l,h}\}_{h\in[H]}\right],\\ &\left[C_l,\ \{M^{l,h}C_l\}_{h\in[H]}\right],\\ &\left[\widehat{W}_g^l, \widehat{W}_u^l,\ \{M^{l,h}\widehat{W}_g^l, M^{l,h}\widehat{W}_u^l\}_{h\in[H]}\right].\end{aligned} \tag{26}$$

Choose $E = C_0P_0^\dagger$. The same induction as in Lemma E.1 first gives $\tilde{A}^{l,h} = A^{l,h}$. The second block of probes then gives

$$\tilde{Y}^lP_l = X^{l-1}C_l + \sum_{h=1}^{H} A^{l,h}X^{l-1}M^{l,h}C_l = Y^lC_l.$$

For $i = (j-1)s_l + b$, the third block gives

$$\tilde{Y}^lR_g^l[:,i] = Y^lW_g^l[:,j], \qquad \tilde{Y}^lR_u^l[:,i] = \alpha_{l,j,b}Y^lW_u^l[:,j].$$

Writing $z_{l,j} := \sigma(Y^lW_g^l[:,j]) \odot (Y^lW_u^l[:,j])$, we obtain

$$\begin{aligned}\tilde{X}^lP_l &= Y^lC_l + \sum_{j=1}^{r} z_{l,j}\alpha_{l,j}B_{l,j}\\ &= Y^lC_l + \sum_{j=1}^{r} z_{l,j}W_d^l[j,:]C_l = X^lC_l.\end{aligned}$$

This proves the residual version of the probe lemma, including uniformity over context length and input.

We now give a deterministic binary witness. For any ordered set $\mathcal{S}$ of coordinates, write $J_{\mathcal{S}}$ for the matrix whose columns are the corresponding standard basis vectors in that order. Begin with $P_L = U = J_{[d_{\text{in}}]}$, and proceed backward. Suppose $P_l = J_{\mathcal{S}}$ for an ordered set $\mathcal{S} \subseteq [s_0]$ of size s_l. Choose ordered coordinate sets

$$\mathcal{G}, \mathcal{U}, \quad \{\mathcal{Q}_h, \mathcal{K}_h\}_{h\in[H]}, \quad \{\mathcal{S}_h, \mathcal{G}_h, \mathcal{U}_h\}_{h\in[H]}$$

of respective sizes $q_l, q_l, d, d, s_l, q_l, q_l$, all disjoint from one another and from $\mathcal{S}$. To make the assignment deterministic, allocate these sets in the displayed order using the smallest unused coordinates of $[s_0]$, with increasing order within each set. This is possible because the total number of coordinates, including $\mathcal{S}$, is exactly $s_{l-1} \le s_0$.

Set $R_g^l = J_{\mathcal{G}}$, $R_u^l = J_{\mathcal{U}}$, $R_Q^{l,h} = J_{\mathcal{Q}_h}$, and $R_K^{l,h} = J_{\mathcal{K}_h}$. For each head, define

$$R_V^{l,h} := J_{\mathcal{S}_h}P_l^\top + J_{\mathcal{G}_h}J_{\mathcal{G}}^\top + J_{\mathcal{U}_h}J_{\mathcal{U}}^\top. \tag{27}$$

The disjoint domains imply

$$R_V^{l,h}P_l = J_{\mathcal{S}_h}, \qquad R_V^{l,h}R_g^l = J_{\mathcal{G}_h}, \qquad R_V^{l,h}R_u^l = J_{\mathcal{U}_h}.$$

Consequently, (25) consists of s_{l-1} distinct coordinate columns and defines the next selector P_{l-1}. Set $R_d^l[I_{l,j},:] = P_l^\top$ for every j. Then every local matrix $B_{l,j}$ is I_{s_l}. The construction is consistent across layers because their parameters are untied.

All parameters are binary. Each value matrix has $s_l + 2q_l$ nonzero entries, each gate, up, or down matrix has q_l, each query or key matrix has d, and U has d_{in}. The width recurrence bounds all these counts by m. At the end, P_0 selects s_0 distinct coordinates from $[s_0]$, so its first s_0 rows form a permutation matrix. This proves the sparse assertion via the residual probe conditions.

Finally, use the determinant polynomial (21) with the residual probes and widths, again fixing the first s_0 rows for $\overline{P}_0$. The recursion (25) makes every probe entry polynomial in the universal parameters. At the deterministic assignment above, $\det(\overline{P}_0) = \pm 1$ and every local determinant is 1. Hence the polynomial is nonzero. The measure-zero argument from Theorem E.3 now proves the almost-sure assertion, on a single event that is independent of the target transformer. □

Scope. These results concern bias-free blocks without normalization and with untied universal parameters across layers. The random theorems use the displayed layer-dependent hidden widths $q_l = rs_l$. The sparse constructions may be padded to a common larger feed-forward width by adding units with zero gate, up, and down parameters.

F Lower bound on embedding dimension

We prove a lower bound for our approach even when all the attention matrices are matched.

Assumption F.1. *For each attention head in the random/designed transformer, the attention matrix is equal to that of the target transformer, i.e. $\tilde{A}^{l,h} = A^{l,h}$, and $\forall l \in [L], h \in [H]$, $A^{l,h}$ is arbitrary row-stochastic matrix, i.e. each row of $A^{l,h}$ sum up to one.*

Theorem F.2 (Lower bound assuming matching attention)**.** *Let $d_{\text{in}} = 1$. Suppose fixed matrices $R_V^{l,h} \in \mathbb{R}^{m\times m}$, $l \in [L]$, $h \in [H]$, have the following property for every context length $n \in \mathbb{N}$. For any target transformer with attention matrices $\{A^{l,h}\}_{l,h}$ and value and output matrices $\{W_V^{l,h}, W_O^{l,h}\}_{l,h}$, there exist $E \in \mathbb{R}^{d_{\text{in}}\times m}$ and $U \in \mathbb{R}^{m\times d_{\text{in}}}$ such that $Z = \tilde{X}^L U$ equals $Y = X^L$ for every input $X^0 \in \mathbb{R}^{n\times d_{\text{in}}}$. Then $m \geq H^{L/2}$.*

Proof. We prove by contradiction. Recall that

$$X^L = \sum_{(h_1,\ldots,h_L)\in[H]^L} A^{L,h_L}\cdots A^{1,h_1} X^0 W_V^{1,h_1} W_O^{1,h_1} \cdots W_V^{L,h_L} W_O^{L,h_L}$$
$$\tilde{X}^L = \sum_{(h_1,\ldots,h_L)\in[H]^L} \tilde{A}^{L,h_L}\cdots \tilde{A}^{1,h_1} X^0 E R_V^{1,h_1} \cdots R_V^{L,h_L}$$

Assuming the attention matrices are the same, for $\tilde{X}^L U = X^L$,

$$\sum_{(h_1,\ldots,h_L)\in[H]^L} A^{L,h_L}\cdots A^{1,h_1} X^0 \left(\prod_{l=1}^{L} W_V^{l,h_l} W_O^{l,h_l} - E\prod_{l=1}^{L} R_V^{l,h_l} U\right) = 0 \tag{28}$$

Let $\mathbf{h} := (h_1, \ldots, h_L)$ and

$$\alpha_{\mathbf{h}} := \prod_{l=1}^{L} W_V^{l,h_l} W_O^{l,h_l} - E\prod_{l=1}^{L} R_V^{l,h_l} U,$$
$$A_{\mathbf{h}} := A^{L,h_L}\cdots A^{1,h_1}.$$

The embedding and unembedding matrices may depend on the target attention matrices. We therefore use a single collection of attention matrices for which output equality forces all coefficients $\alpha_{\mathbf{h}}$ to vanish simultaneously.

Lemma F.3. *Let $N := H^L$ and $n := 2N$. There exist $n \times n$ row-stochastic matrices $\{A^{l,h}\}_{l\in[L],h\in[H]}$ and an input $X^0 \in \mathbb{R}^{n\times 1}$ such that the N vectors*

$$\left\{A^{L,h_L}\cdots A^{1,h_1} X^0 : (h_1,\ldots,h_L) \in [H]^L\right\}$$

are linearly independent.

Proof. Index N blocks of size 2 by $s = (s_1, \ldots, s_L) \in [H]^L$. Let

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}, \qquad e_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

For each $l \in [L]$ and $h \in [H]$, define the block-diagonal matrix

$$A^{l,h} := \bigoplus_{s \in [H]^L} A_s^{l,h}, \qquad A_s^{l,h} := \begin{cases} I, & h = s_l, \\ B, & h \neq s_l. \end{cases}$$

These matrices are row-stochastic. Take X^0 to be the column vector obtained by stacking N copies of e_1, in the same block order.

Fix a path $\mathbf{h} = (h_1, \ldots, h_L)$. In block $s = \mathbf{h}$, every factor in $A^{L,h_L} \cdots A^{1,h_1}$ equals I. In every other block, at least one factor equals B. Since

$$IB = BI = BB = B, \qquad Ie_1 = e_1, \qquad Be_1 = 0,$$

the block indexed by s in $A_{\mathbf{h}} X^0$ is

$$(A_{\mathbf{h}} X^0)_s = \begin{cases} e_1, & s = \mathbf{h}, \\ 0, & s \neq \mathbf{h}. \end{cases}$$

Thus the N path vectors are nonzero and have disjoint supports, so they are linearly independent. □

Fix the single collection of attention matrices and input from this lemma. For any target value weights, choose E, U as in the theorem for this collection. In the block indexed by $\mathbf{h}$, (28) becomes $\alpha_{\mathbf{h}} e_1 = 0$. Consequently, $\alpha_{\mathbf{h}} = 0$ for every $\mathbf{h} \in [H]^L$, without changing the attention matrices or the chosen E, U. Next, we show that if $m < H^{L/2}$, there are target value weights for which this is impossible.

Lemma F.4. *If $m < H^{L/2}$, then for each collection of $m \times m$ matrices $\{R^{l,h}\}_{(l,h) \in [L] \times [H]}$, there exists a collection of $d_{\text{in}} \times d_{\text{in}}$ matrices $\{W^{l,h}\}_{(l,h) \in [L] \times [H]}$ such that, for each pair of $d_{\text{in}} \times m$ matrices E, U, there exists $(h_1, \ldots, h_L) \in [H]^L$ such that*

$$E \prod_{l=1}^{L} R^{l,h_l} U^\top \neq \prod_{l=1}^{L} W^{l,h_l}.$$

Proof. Fix a collection of $m \times m$ matrices $\{R^{l,h}\}_{(l,h) \in [L] \times [H]}$. Consider the system of H^L linear equations $Ax = b$ in m^2 unknowns x, where the equation corresponding to $(h_1, \ldots, h_L) \in [H]^L$ is

$$\operatorname{tr}\left(\prod_{l=1}^{L} R^{l,h_l} \operatorname{mat}(x) \right) = \prod_{l=1}^{L} W^{l,h_l}.$$

Here, $\operatorname{mat}(x)$ reshapes x to a $m \times m$ matrix. The matrix A is determined by $\{R^{l,h}\}_{(l,h) \in [L] \times [H]}$; below, we show how to choose $\{W^{l,h}\}_{(l,h) \in [L] \times [H]}$, which in turn determines the right-hand side vector b.

The rank of A is at most the number of columns, which is m^2; that number is at most $H^L - 1$ by the assumption $m < H^{L/2}$. So by the rank-nullity theorem, the left nullspace of A has dimension at least 1. Let y be a nonzero vector in the left nullspace of A, and let $y_{h_1^*, \ldots, h_L^*}$ be a nonzero component of y. For each $(l, h) \in [L] \times [H]$, choose

$$W^{l,h} := \begin{cases} 1 & \text{if } h = h_l^*; \\ 0 & \text{otherwise;} \end{cases}$$

and let b be the corresponding right-hand side vector. We have

$$\langle y, b \rangle = \sum_{(h_1, \ldots, h_L) \in [H]^L} y_{h_1, \ldots, h_L} \times \prod_{l=1}^{L} W^{l,h_l} = y_{h_1^*, \ldots, h_L^*} \times 1 \neq 0.$$

Therefore, b has a nonzero component in this left nullspace, which implies that b is not in the column space of A, i.e., $Ax = b$ has no solution. Since $U^\intercal E$ is a $m \times m$ matrix, it follows that there exists some $(h_1, \ldots, h_L) \in [H]^L$ such that

$$E \prod_{l=1}^{L} R^{l,h_l} U^\intercal = \operatorname{tr}\left(\prod_{l=1}^{L} R^{l,h_l} U^\intercal E \right) \neq \prod_{l=1}^{L} W^{l,h_l}$$

□

The proof of Theorem F.2 follows from this. □

The lower bound also works for the weight-tying (looped) transformer, by choosing

$$W^h := \begin{cases} 1 & \text{if } h = h_l^*; \\ 0 & \text{otherwise.} \end{cases}$$

G Additional ablation studies

Besides parenthesis balancing and 2-hop induction heads, we also perform ablation studies on 1-hop, 3-hop, and 4-hop induction heads tasks. These experiments follow the same protocol as in the parenthesis balancing and 2-hop induction heads: for the sparse and random universal transformers, only the embedding and unembedding matrices are trained, while all internal attention parameters are kept fixed. When MLP layers are added, they are randomly initialized and also kept fixed. The results are shown in Tables 3 and 4. These results are consistent with Table 2b, that while it's hard for the attention-only universal transformers to find the optimal solution, adding residual connections and layer normalization can make the optimization much easier.

Table 3: Induction head (1-hop) results for $H = 2$, $L = 2$, head_dim $= 28$, embedding_dim $= 456$.

Model	Sparse	Random	Fully-trained
Attention-only	30.2%	29.4%	8.4%
+ residual	42.1%	5.5%	67.5%
+ residual and LN	100%	96.1%	100%
+ MLP	61.0%	51.3%	100%
+ MLP and LN	100%	96.0%	100%

Table 4: 3-hop and 4-hop results for $H = 2$, $L = 4$, head_dim $= 30$, embedding_dim $= 2280$.

3-hop

Model	Sparse	Random	Fully-trained
Attention-only	39.2%	32.6%	99.5%
+ residual	53.6%	43.9%	99.9%
+ residual and LN	100%	92.6%	100%
+ MLP	56.9%	57.5%	100%
+ MLP and LN	100%	98.7%	100%

4-hop

Model	Sparse	Random	Fully-trained
Attention-only	39.4%	35.2%	95.6%
+ residual	43.9%	41.6%	97.9%
+ residual and LN	100%	89.2%	100%
+ MLP	44.7%	48.8%	97.4%
+ MLP and LN	82.8%	70.9%	100%

H Experimental details

H.1 Data generation

Parenthesis balancing. The data is generated following the same way as [29]. We generate sequences over a four-symbol vocabulary consisting of padding/EOF, left parenthesis, right parenthesis, and a query token, encoded as $0, 1, 2, 3$, respectively. Each example is constructed by first sampling a parenthesis string of length at most $2\,\texttt{max_len}$. With probability $1/3$, we sample a completely random parenthesis string of uniformly random length; otherwise, we recursively generate a balanced parenthesis string with a uniformly sampled number of matching pairs. To obtain both positive and negative instances, the sampled string is optionally corrupted by local mutations. These mutations include swapping two positions in the string and flipping parentheses from left to right or right to left, with a geometrically distributed number of repetitions. After the parenthesis string s is generated, we compute whether it is balanced using the standard stack-depth criterion: the running depth must never become negative and must end at zero. We then append a query token "?" followed by the target answer token. The answer is ")" if s is balanced and "(" otherwise. Finally, the sequence is padded with zeros to a fixed length $2\,\texttt{max_len} + 3$. The model is trained in a next-token prediction format: for each padded sequence, the input is the sequence excluding the final token, and the target is the same sequence shifted one position to the left.

k-hop induction heads. For this task, we generate the data following [21]. We generate sequence-modeling examples over a vocabulary consisting of character tokens, induction-query tokens, a blank token, and a special "does-not-exist" token. The character vocabulary contains `char_tokens` symbols, denoted by consecutive lowercase letters. For each example, we first sample a random character string $x_1, \ldots, x_{\texttt{seq_len}}$, where the first character is sampled uniformly and each subsequent character is sampled uniformly subject to being different from the previous character. This prevents trivial adjacent repetitions. We then construct a sequence of induction-hop targets. The 0-hop string is the original string itself. Given the current hop string and its associated source indices, the next-hop string is obtained positionwise as follows: for position i, we look backward in the original random string for the most recent previous occurrence of the current symbol before its current source index. If such an occurrence exists, the next-hop target is the character immediately following that occurrence; otherwise, the target is the special "does-not-exist" symbol. Repeating this procedure produces targets for all hop depths up to `max_hops`. For each training example, the queried hop depth is either fixed or sampled uniformly from $\{\texttt{min_hops}, \ldots, \texttt{max_hops}\}$ and encoded as a leading induction-query token. The model input consists of this query token followed by the random character string with the last two characters removed. The output is a blank token followed by the corresponding hop-target string, also truncated by two characters. Thus the task is trained in a full-sequence prediction format: after reading the hop query and the character context, the model must output, at every position, the result of applying the specified number of induction steps to the underlying sequence.

H.2 Training details

All the experiments are conducted on one single NVIDIA RTX PRO 6000 Blackwell 96GB. This project roughly takes 14 GPU days.

The embedding and unembedding matrices are initialized from standard normal distribution with standard deviation 0.02. We use AdamW as optimizer and LambdaLR as learning rate scheduler. We adopt online training, i.e. at each step we generate fresh data.

The hyperparameters for parenthesis balancing and k-hop induction heads task can be found in Tables 5 and 6. For $k = 1, 2, 3, 4$, the training steps are $10000, 15000, 60000, 60000$ respectively.

Hyperparameter	Value
Embedding dimension m	1024
Head dimension d	24
Depth L	2
Number of heads H	4
Vocabulary size	4
Activation function of MLP	GeLU
Layer norm ϵ	10^{-5}
Learning rate	10^{-3}
Warmup steps	50
Training steps	10^4
Batch size	1000

Table 5: Model and training hyperparameters for parenthesis balancing.

Hyperparameter	Value
Number of heads H	2
Number of character tokens	4
Vocabulary size	30
Activation function of MLP	GeLU
Layer norm ϵ	10^{-5}
Learning rate	10^{-4}
Warmup steps	1000
Batch size	128

Table 6: Model and training hyperparameters for k-hop induction heads.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state the assumptions and the setup for our results in the introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss our limitations in the conclusion and future work section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide all the assumptions in the theorem statements and all the proof in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the setup and exact model size in the main paper and hyperparameters in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We need to clean up the code, and will release in the final version.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify all of them in the experimental details section in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We plot the standard deviation and maximum minimum accuracy in the error bars.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include this in the experimental detail section of the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper is mainly theoretical and does not involve any experiments in the human level.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This paper is mainly a theoretical paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The experiments involved are only using synthetic data for toy models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We clearly state which papers we are using to generate our training data and code.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve any research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve any research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: We only use LLM for drafting and editing the paper.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.