ON SURJECTIVITY OF NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Given a trained neural network, can any specified output be generated by some input? Equivalently, does the network correspond to a function that is surjective? In generative models, surjectivity implies that any output, including harmful or undesirable content, can in principle be generated by the networks, raising concerns about model safety and jailbreak vulnerabilities. In this paper, we prove that many fundamental building blocks of modern neural architectures, such as networks with pre-layer normalization and linear-attention modules, are almost always surjective. As corollaries, widely used generative frameworks, including GPT-style transformers and diffusion models with deterministic ODE solvers, admit inverse mappings for arbitrary outputs. By studying surjectivity of these modern and commonly used neural architectures, we contribute a formalism that sheds light on their unavoidable vulnerability to a broad class of adversarial attacks.

1 INTRODUCTION

Deep generative models have achieved remarkable success in recent years—spanning natural language processing (OpenAI et al., 2024; Touvron et al., 2023; Chowdhery et al., 2023), computer vision (Imagen-Team-Google et al., 2024; Grattafiori et al., 2024), and robotics (Kim et al., 2024; Team et al., 2025). Yet, despite this progress, we still understand little about their *global* input–output structure. In many applications, one would like the output of a learned model to satisfy hard constraints or remain within a restricted region of the output space: for example, constraints in safe reinforcement learning and control (Achiam et al., 2017; Dalal et al., 2018), or constrained sequence generation at decoding time (Hokamp & Liu, 2017; Post & Vilar, 2018). Training objectives can encourage such behavior, but they rarely provide rigorous guarantees. Understanding whether such constraints can hold fundamentally requires characterizing the full range of outputs a model can produce. A natural mathematical lens for this question is *surjectivity*. A function $f : \mathcal{X} \to \mathcal{Y}$ is *surjective* if for every $y \in \mathcal{Y}$, there exists an $x \in \mathcal{X}$ such that $f(x) = y$. Viewing a neural network with parameters θ as a function f_θ from an input space to an output space, surjectivity asks the global question: *can any specified output be generated by some input?* Motivated by this perspective, we study surjectivity as a structural property of trained generative models.

Formalizing surjectivity for *trained* neural networks surfaces challenges and viewpoints that are not common in standard learning theory. On the one hand, it is clearly too much to expect that *every* choice of parameters yields a surjective function: pathological settings (e.g., all weights set to zero) can collapse the network to a constant map. On the other hand, merely exhibiting *some* parameters that yield surjectivity is also uninformative, since there is no guarantee that training will discover them. We also seek to understand this global property of trained models in a way that is robust to variations in data, optimization, and the choice of loss functions, with the goal of capturing surjectivity of modern architectures at a broad, structural level.

To address this, we adopt a probabilistic formalism that matches the practical reality that trained parameters are the outcome of a complex pipeline. Rather than asking whether surjectivity holds for *all* or *some* parameter settings, we ask whether it holds *almost always*: for a fixed architecture, do all except for a measure-zero subset of parameter configurations yield a surjective function? This formalism is intentionally agnostic to the precise training procedure. The evolution of parameters depends intricately on optimizer choice, data distribution, loss function, and other aspects of the pipeline, all of which introduce randomness. If surjectivity holds almost everywhere in parameter space, then *regardless of these fine-grained details*, it is exceedingly unlikely that the resulting trained model is not surjective.

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

Technical Results and Toolkit: Differential Topology Given the significant departure from existing paradigms of studying neural networks, our perspective also calls for a toolkit that can reason about global structure. We show that *differential topology* is the right tool for the job. Differential topology studies smooth maps between (typically high-dimensional) spaces and provides invariants and existence theorems that connect *local* behavior (e.g., derivatives and Jacobians) to *global* consequences (e.g., whether points must have pre-images). This connection is far from accidental: modern networks are built from (piecewise) smooth components precisely to enable optimization via backpropagation, which makes them naturally amenable to differential-topological analysis. In particular, we rely on the intermediate value theorem and Brouwer's fixed point theorem. These results let us prove the existence of pre-images by reducing the target equation to a univariate polynomial equation or a fixed point problem.

Using these tools, we analyze the surjectivity of core building blocks in modern architectures, including LayerNorm with residual connections, and attention mechanisms. In particular, we show that any continuous function wrapped with Pre-LayerNorm is surjective (Theorems 3.1 and 3.2)—implying that both Attention and MLP layers are surjective when wrapped in Pre-LayerNorm. We also establish that linear attention (retention, Theorem 3.4) is almost always surjective. Two notable exceptions exist: we show that (softmax) Attention itself and MLP with ReLU activation are not surjective.

Architecture	Surjectivity
MLP with ReLU/GeLU	✘
MLP with Pre-LayerNorm	✔
Attention	✘
Attention with Pre-LayerNorm	✔
Linear Attention (Retention)	✔

Table 1: Partial Summary of Results in Section 3. Wrapping a function f in Pre-LayerNorm is defined with residual connection $f(\mathrm{LN}(x)) + x$.

Broader Implications of Surjectivity: Constraints and Jailbreaks. Surjectivity is a fundamental mathematical property, but it has particular significance for modern generative models. Unlike classification tasks—where failure of surjectivity simply means some labels are unreachable—generative models operate over rich, high-dimensional output spaces where the reachable set itself is central. From this perspective, "almost always" surjectivity can be viewed as a structural limitation: typical training cannot teach a model to "stay within" arbitrary output constraints. In particular, surjectivity guarantees pre-images, and thus inverse mappings, for arbitrary outputs. This is precisely the phenomenon exploited by jailbreak-style attacks, which search for inputs that elicit prohibited or undesired behavior.

In Section 4, we discuss concrete implications in language, vision, and robotics. We show that Transformers, diffusion models with deterministic ODE solvers, and certain robotics policy networks are surjective. These results highlight obstacles to obtaining provably safe behavior purely through training on common architectural families. At the same time, surjectivity is existential and does not guarantee that problematic inputs can be found efficiently or with limited information. We discuss these considerations and their connection to existing attacks, as well as implications for safety, jailbreak robustness, and copyright risk assessment.

2 Notation and Preliminaries

We denote the sets of real and complex numbers by $\mathbb{R}$ and $\mathbb{C}$, respectively. By defualt, we name individual vectors by letters x, y, and sequence vectors using a, b, c. When using subscript, x_i, y_i mean the i-th entries, while a_i, b_i, c_i mean their i-th elements. Symbol $\|\cdot\|$ represents the 2-norm of a vector or matrix. Symbol $\odot$ indicates entry-wise multiplication between vectors or matrices. Symbol $\oplus$ represents the direct sum (Cartesian Product) of linear spaces and we use it to define input spaces for sequences. For a positive integer $n \in \mathbb{N}^+$, we define $[n] = \{1, \cdots, n\}$. The identity matrix of dimension d is represented by I^d. For a set $\Omega \subset \mathbb{R}^d$, let $\partial\Omega$ be its boundary and $\overline{\Omega}$ be its closure. For a function $f : \mathbb{R}^d \to \mathbb{R}^d$ and a set $S \subset \mathbb{R}^d$, we denote $f(S) = \{f(x) | x \in S\}$. *For most of this paper, we only discuss networks with the same input and output dimensions.*

2.1 NEURAL NETWORKS

Next, we give an overview of common modern neural network building blocks. Let the input of the network be a vector $x \in \mathbb{R}^d$ where $d \in \mathbb{N}$ is the input dimension. The output $y \in \mathbb{R}^d$ is also a vector. One of the most elementary architectures is the Multi-Layer Perceptron.

Definition 1. *An* m-layer Multi-Layer Perceptron (MLP) *is a function* $f : \mathbb{R}^d \to \mathbb{R}^d$ *defined as*

$$f(x) = \sigma_m(W_m \cdots \sigma_2(W_2\sigma_1(W_1x + \lambda_1) + \lambda_2) \cdots + \lambda_m)$$

where $\{W_i\}_{i\in[m]}$ *are trainable matrices,* $\{\lambda_i\}_{i\in[m]}$ *are trainable vectors called* bias terms*, and* $\{\sigma_i\}_{i\in[m]}$ *are nonlinear entry-wise functions called* activation functions*. Row dimensions of* $W_1, \cdots, W_{m-1}$ *are called* hidden dimensions*.*

We define common examples of activation functions, namely ReLU, and GeLU (Hendrycks & Gimpel, 2023) below:

$$\mathrm{ReLU}(x)_i = \max\{x_i, 0\}, \mathrm{GeLU}(x)_i = x_i \cdot \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{x_i}{\sqrt{2}}\right)\right].$$

Here the subscript i means the i-th entry of a vector, and $\mathrm{erf}(\cdot)$ is the Gaussian error function.

Residual connection (He et al., 2016) and layer normalization (Ba et al., 2016) are essential elements of modern deep neural networks that usually work together.

Definition 2. Layer Normalization[1] *is a function* $\mathrm{LN} : \mathbb{R}^d \to \mathbb{R}^d$ *defined as*

$$\mathrm{LN}(x) = \gamma \odot \frac{x - \overline{x}}{\|x - \overline{x}\|/\sqrt{d}} + \beta, \quad \overline{x} = \frac{1}{d}\sum_{i=1}^{d} x_i$$

where $\gamma, \beta \in \mathbb{R}^d$ *are trainable parameters. When used with residual connections, there are two variants called* Pre-LayerNorm *and* Post-LayerNorm*. When wrapped around a function* $f : \mathbb{R}^d \to \mathbb{R}^d$*, residual connection, Pre-LayerNorm and Post-LayerNorm are respectively defined as*

$$g(x) = f(x) + x, \quad g(x) = f(\mathrm{LN}(x)) + x, \quad g(x) = \mathrm{LN}(f(x) + x).$$

Pre-LayerNorm has become common practice in modern neural networks as it stabilizes training (Xiong et al., 2020; OpenAI et al., 2024; Touvron et al., 2023; Chowdhery et al., 2023). Post-LayerNorm is also used sometimes (Zhuo et al., 2025; Li et al., 2024; Vaswani et al., 2017).

Sequence Models. There are also specialized architectures dealing with sequential data, which take in an input sequence $a_1, \cdots, a_n$ and output another sequence $b_1, \cdots, b_n$. Attention (Bahdanau et al., 2016; Vaswani et al., 2017) is one of the most widely used sequence models.

Definition 3. *For trainable key, query, and value matrices* $K, Q, V \in \mathbb{R}^{d\times d}$*, a* causally-masked attention layer *calculates outputs as*[2]

$$b_i = \mathrm{Attn}(a)_i = \frac{1}{Z_i}\sum_{j=1}^{i} \exp\big(a_j^\top K^\top Q a_i\big) V a_j, \quad Z_i = \sum_{j=1}^{i} \exp\big(a_j^\top K^\top Q a_i\big).$$

Causally-masked attention layers are usually used for autoregressive generation. Specifically, given input $a_1, \cdots, a_n$, the next token is generated from decoding b_n. After that this token is appended to the input and subsequent tokens are iteratively generated in the same way. There are also variants to Attn called linear attentions (Yang et al., 2024). The simplest linear attention is RetNet (Sun et al., 2023).

Definition 4. *For trainable parameters* $K, Q, V \in \mathbb{R}^{d\times d}$*, a* Retention layer *calculates output as*

$$b_i = \mathrm{Ret}(a)_i = \sum_{j=1}^{i} \left(a_j^\top K^\top Q a_i\right) V a_j = S_i Q a_i, \text{ where } S_i = S_{i-1} + V a_i a_i^\top K^\top. \tag{1}$$

[1] In real-world implementation a small ε is added to the denominator in LN. Here we omit it for presentation simplicity and it does not change the proofs in this paper.

[2] In practice, a scaling factor is inside the exp function (Vaswani et al., 2017), which we omit, as it does not affect our analysis.

In other words, Ret can be thought of as Attn without the non-linearity introduced by the soft-max function through $\exp$ and Z_i. It admits a recurrent form as shown in Equation (1), so autoregressive generation becomes faster. Other variants keep the recurrent form and use more complicated update rules for better performance. To keep the presentation clean, we defer the discussions on multi-head attention to the Appendix.

Transformer. The Transformer architecture is widely used in many applications recently. Most of them can be expressed by the building blocks stated previously. To illustrate this, we take GPT-3 (Brown et al. (2020) referred to as GPT below) as an example here. A single block in GPT can be expressed as

$$b_i = \mathrm{TF}(a)_i = W_2 \mathrm{GeLU}(W_1 \mathrm{LN}\left(c_i\right) + \lambda_1) + \lambda_2 + c_i, \text{ where } c_i = \mathrm{Attn}(\mathrm{LN}(a))_i + a_i.$$

Matrices $W_1, W_2^\top \in \mathbb{R}^{d \times d'}$. Besides, LN applying to a sequence means applying separately to each input vector, i.e. the layer norm of the i-th vector is $\mathrm{LN}(a)_i = \mathrm{LN}(a_i)$. In plain text, TF is an Attn followed by a two-layer MLP, each wrapped with Pre-LayerNorm, and GPT is a composition of several TFs.

2.2 Differential Topology

Differential topology studies the properties of smooth manifolds that are invariant under smooth transformations. Since almost all neural networks are trained using back-propagation, the architectures are usually smooth.[3] Differential topology hence provides natural tools for us to prove surjectivity of neural networks. In this section, we go over the necessary mathematical concepts and results that will be used later. We restrict our scope to smooth maps $f, g : \mathbb{R}^d \to \mathbb{R}^d$ here unless otherwise specified

We start with the intermediate value theorem, a familiar result from calculus that follows from the connectedness of real intervals, a topological property.

Theorem 2.1 (Intermediate Value Theorem)**.** *Let $a < b$ be real numbers and let $f : [a, b] \to \mathbb{R}$ be continuous. For every y between $f(a)$ and $f(b)$, there exists $x \in [a, b]$ such that $f(x) = y$.*

For univariate polynomials, this gives a simple surjectivity guarantee.

Corollary 2.2. *Every univariate polynomial $p : \mathbb{R} \to \mathbb{R}$ of odd degree with real coefficients is surjective. In particular, p has a real root.*

At a high level, this is because for any real number $y \in \mathbb{R}$ the leading term for $p(x) - y$ takes opposite signs for sufficiently large positive and negative inputs. By Theorem 2.1, there exists $x \in \mathbb{R}$ such that $p(x) - y = 0$, and hence p is surjective. We give a formal proof in Section B.

One of the early triumphs of topology is the Brouwer's fixed point theorem.

Theorem 2.3 (Brouwer's Fixed Point Theorem, (Munkres, 1984, p. 117))**.** *Let $B^d(R) = \left\{x \in \mathbb{R}^d \middle| \|x\| \le R\right\}$ be a d-dimensional ball with radius R. For every continuous function $f : B^d(R) \to B^d(R)$, there exists $x \in B^d(R)$ such that $f(x) = x$.*

This theorem can be generalized from a ball $B^d(R)$ to any convex closed bounded set. Most celebrated and common applications of Brouwer's fixed point theorem are in game theory, Economics, and the study of equilibria of dynamical systems.

Definition 5. Differential $Df : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ *is defined as* $Df(x)_{ij} = \partial f_i(x)/\partial x_j$.

Since $f(x)$ can be approximated linearly in a small neighborhood around x, $Df(x)$ describes the local behavior of f around x. More specifically $Df(x)_{ij}$ describes the rate the i-th dimension of output changes with regard to the j-th dimension of input. Hence an invertible $Df(x)$ (equivalently one with $\det Df(x) \neq 0$) indicates that f behaves well around x in the sense that no small neighborhood containing x is collapsed to be lower dimensional after the application of function f. The next theorem, known as the Inverse Function Theorem, formalizes this intuition.

[3] Some building blocks, such as ReLU are not smooth everywhere. However they are only not smooth on zero measure sets, otherwise we cannot obtain gradient for a substantial amount of inputs. Topology can still tackle this scenario because such functions can always be approximated by smooth functions. We refer interested readers to standard references (Hirsch, 1976) and omit this subtlety for simplicity.

Theorem 2.4 (Inverse Function Theorem). *Let $x \in \mathbb{R}^d$ satisfy $\det Df(x) \neq 0$, then there exist open sets $U \ni x, V \ni f(x)$, such that f is bijective between U and V.*

3 Surjectivity of Architectural Blocks in Modern Neural Networks

In this section, we analyze the surjectivity of fundamental building blocks in modern neural networks using tools from differential topology. Before diving into the details, we formalize the setting. Though elementary, we start with the formal definition of surjectivity.

Definition 6. *A function $f : \mathcal{X} \to \mathcal{Y}$ is* surjective*, if for any $y \in \mathcal{Y}$, there exists a pre-image, i.e., an $x \in \mathcal{X}$ such that $f(x) = y$.*

Surjectivity is closed under composition. Namely, if $f : \mathcal{X} \to \mathcal{Y}, g : \mathcal{Y} \to \mathcal{Z}$ are surjective, their composition $g \circ f$ is also surjective. This allows us to separately prove surjectivity of building blocks of neural networks and conclude that the whole network is surjective. It is often too good to hope that an architecture is always surjective with any parameter. For example, if we set $V = 0$ in attention (Definition 3), this layer will output nothing but zero. A less extreme example is when V is not an invertible matrix, it is not possible to output a vector outside the subspace spanned by V's column space. However, this almost never happens in practice because the set of non-invertible matrices takes up zero volume in the parameter space, and hence is almost never hit in trained models.

Definition 7. *Let $\mathcal{H}_\Theta = \{f_\theta : \mathcal{X} \to \mathcal{Y} | \theta \in \Theta\}$ be a class of neural networks parameterized by $\theta \in \Theta$, where Θ is a subset of Euclidean space with Lebesgue measure and $\mathcal{Y} \subset \mathbb{R}^d$ has Lebesgue measure. $\mathcal{H}_\Theta$ is an* almost always surjective *set if, except for θ in a zero-measure subset of Θ, $f_\theta^{-1}(y)$ is nonempty for every $y \in \mathcal{Y}$ except for a zero-measure subset of $\mathcal{Y}$.*

Below we analyze which architectures are almost always surjective and in Section 4 we discuss which real-world models are surjective when used in specific ways.

3.1 Pre-LayerNorm

In this section, we prove that Pre-LayerNorm is surjective using a neat but nontrivial application of the Brouwer's fixed point theorem (Theorem 2.3).

Theorem 3.1. *Let $f : \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function, then $g : \mathbb{R}^d \to \mathbb{R}^d$ defined by $g : x \mapsto f(\mathrm{LN}(x)) + x$ is surjective.*

Proof. By the definition of Pre-LayerNorm, the normalized vector before the affine transformation has 2-norm $\sqrt{d}$. Therefore, $\|\mathrm{LN}(x)\| \leq \sqrt{d}\|\gamma\| + \|\beta\|$. Since f is continuous on the compact ball of radius $\sqrt{d}\|\gamma\| + \|\beta\|$, we have

$$M = \sup_{x \in \mathbb{R}^d} \|f(\mathrm{LN}(x))\| < \infty.$$

To prove surjectivity of g, we need to prove that for any $y \in \mathbb{R}^d$ there exists an input x^* such that $g(x^*) = y$. Below we fix y and prove such x^* exists. Let $F : \mathbb{R}^d \to \mathbb{R}^d$ be a function defined by $F(x) = y - f(\mathrm{LN}(x))$. Now we find a fixed point for F. Let $R = M + \|y\| + 1$, then by triangle inequality we have $\|F(x)\| < R$. Therefore, F maps $B^d(R)$ into itself. We can thus define the restriction of F on $B^d(R)$ as a function $F|_{B^d(R)} : B^d(R) \to B^d(R)$. By Theorem 2.3, we know that there exists $x^* \in B^d(R)$ such that $F|_{B^d(R)}(x^*) = x^*$. Plugging this x^* back to the definition of F we know $F(x^*) = x^* = y - f(\mathrm{LN}(x^*))$. Thus $g(x^*) = f(\mathrm{LN}(x^*)) + x^* = y$. This establishes that for any $y \in \mathbb{R}^d$ there exists a corresponding input x^*, so g is surjective. □

This is a powerful theorem in the sense that it places minimal requirements on the function f. Nearly all modern neural networks are continuous, so the theorem implies that any architecture wrapped with Pre-LayerNorm is surjective. Notably, the proof does not rely on the specific expression of LayerNorm, but only on the fact that it is continuous and bounded. Hence if one uses other types of normalization functions with this property instead, like RMSNorm (Zhang & Sennrich, 2019), GroupNorm (Wu & He, 2018) or DyT (Zhu et al., 2025), this theorem still holds. Finally, this theorem can be easily extended to sequence models as we show below.

Theorem 3.2. *Let $f : \oplus_{i\in[n]}\mathbb{R}^d \to \oplus_{i\in[n]}\mathbb{R}^d$ be a continuous function, then $g : \oplus_{i\in[n]}\mathbb{R}^d \to \oplus_{i\in[n]}\mathbb{R}^d$ defined by $g(a)_i = f(\mathrm{LN}(a))_i + a_i$ is surjective.*

The proof of Theorem 3.2 is deferred to the Appendix.

Next, we briefly discuss Post-LayerNorm for completeness. Since LN normalizes its input, Post-LayerNorm cannot be onto $\mathbb{R}^d$; ignoring constant inputs where the idealized LN without ε is undefined, its image is

$$S = \left\{ \gamma \odot z + \beta \middle| z \in \mathbb{R}^d,\ \|z\| = \sqrt{d},\ \overline{z} = 0 \right\}.$$

It suffices for $f(x) + x$ to cover a neighborhood of 0: any nonconstant pre-image of $y \in S$ can be scaled into that neighborhood without changing its LayerNorm.

Theorem 3.3. *Let $f : \mathbb{R}^d \to \mathbb{R}^d$ be a bias-free GeLU MLP whose hidden dimensions are all d. Then $\mathrm{LN}(f(x) + x)$ is almost always surjective on S.*

Proof. Let $g(x) = f(x) + x$ and $s = \mathrm{GeLU}'(0)$. By the chain rule,

$$Dg(0) = sI^d \cdot W_m \cdot sI^d \cdots sI^d \cdot W_1 + I = s^m \prod_{i=m}^{1} W_i + I$$

which is almost always full rank. Since $g(0) = 0$, Theorem 2.4 gives an open $U \ni 0$ with $U \subset g(\mathbb{R}^d)$. For any $y \in S$, choose nonconstant v with $\mathrm{LN}(v) = y$ and $\mu > 0$ with $\mu v \in U$. Then $\mathrm{LN}(\mu v) = y$, and since $\mu v = g(x)$ for some x, we have $\mathrm{LN}(g(x)) = y$. □

Remark 1. We do not aim to provide a comprehensive result for MLPs with Post-LayerNorm. Many of the conditions in this proposition can be relaxed. The network architecture can be quite flexible, as long as its determinant almost never vanishes and the zero input maps to zero output. Moreover, MLPs without bias terms are not uncommon in practice (Groeneveld et al., 2024; Touvron et al., 2023; Chowdhery et al., 2023).

This S-surjectivity does not compose directly, since the next block's domain would also become S; we leave S-to-S Post-LayerNorm surjectivity for future work.

3.2 Linear Attention

In recent years, linear attention mechanisms have gained significant attention due to their improved scalability compared to the original attention mechanism. In this section, we prove that the Retention layer (Definition 4) is almost always surjective.

Theorem 3.4. Ret *is almost always surjective.*

Before presenting the proof, we introduce some notation. As before, the main task is to find an input that produces a prescribed output. Since Retention is autoregressive, it is natural to study surjectivity one position at a time. In particular, b_j depends only on $a_1, \ldots, a_j$. Once we have chosen an input prefix $a_1, \ldots, a_m$ to reach the first m target outputs, we can vary a_{m+1} without changing those outputs. The next step is to solve for a_{m+1} given a target b_{m+1}. Separating the contribution of the prefix from that of the current input gives

$$b_{m+1} = S_m Q a_{m+1} + (a_{m+1}^\top K^\top Q a_{m+1}) V a_{m+1}, \qquad S_m = \sum_{i=1}^{m} V a_i a_i^\top K^\top.$$

To simplify the expression, suppose V is invertible[4] and put

$$x_i = V a_i, \qquad N = V^{-\top} K^\top Q V^{-1}, \qquad X = [x_1, \ldots, x_m], \qquad C = X X^\top, \tag{2}$$

where $V^{-\top} = (V^{-1})^\top$. Write $x = V a_{m+1}$ for the current input in these coordinates. For an empty prefix, $m = 0$ and $S_0 = C = 0$. The next-token map then becomes

$$b_{m+1} = CNx + (x^\top N x)x =: F_{N,X}(x). \tag{3}$$

The proof proceeds in two steps. First, we establish conditions under which $F_{N,X}$ reaches almost every target output. We then show that these conditions almost always hold.

[4]Singular matrices V form a measure-zero set by Lemma 3, since $\det V$ is a nonzero polynomial in the entries of V. We may exclude this set when studying almost always surjectivity.

The technical difficulties. Equation (3) reveals two technical difficulties. First, the coefficient $C = XX^\top$ depends on the preceding inputs. These inputs were chosen to reach the earlier target outputs, and there is no immediate guarantee that they satisfy the conditions needed to solve the next-token equation. For example, we cannot assume that C is invertible in the same way as V: since $\mathrm{rank}(C) \leq m$, C is always singular when $m < d$. We must therefore establish suitable conditions on both N and X under which $F_{N,X}$ reaches almost every target output, and prove that these conditions fail only on a measure-zero set.[5]

Second, Equation (3) is a multivariate polynomial map. The univariate surjectivity guarantee in Corollary 2.2, obtained from the intermediate value theorem in Theorem 2.1, does not directly apply to a map from $\mathbb{R}^d$ to $\mathbb{R}^d$ when $d > 1$. The particular form of Equation (3) is therefore essential to the argument.

Sufficient conditions for almost always surjectivity. We first state the conditions used in the proof, and then show the first step of the proof in Lemma 1.

Assumption 3.5. *Let $d \geq 1$ and $m \geq 0$ be integers, let $N \in \mathbb{R}^{d\times d}$, and let $X \in \mathbb{R}^{d\times m}$, with X empty when $m = 0$. Then:*

1. *If $m = 0$, then $N + N^\top \neq 0$.*
2. *If $1 \leq m < d$, then $G = X^\top NX$ is invertible, its characteristic polynomial has no repeated root over $\mathbb{C}$, and there is a vector $z \in \ker(X^\top N)$ such that $z^\top Nz \neq 0$.*
3. *If $m \geq d$, then C is positive definite and N is invertible.*

Lemma 1. *Under Assumption 3.5, $\mathbb{R}^d \setminus F_{N,X}(\mathbb{R}^d)$ has measure zero, where $F_{N,X}(x) = CNx + (x^\top Nx)x$. If $m \geq d$, $F_{N,X}$ is surjective onto all of $\mathbb{R}^d$.*

We present a proof sketch for the second case of Assumption 3.5 below and defer the full proof, including the cases $m = 0$ and $m \geq d$, to Section B.3. This case contains the main technical steps.

Proof sketch of Lemma 1 for $1 \leq m < d$. The high level idea of the proof is to reduce finding the root of a vector polynomial to finding a root of a scalar polynomial, and apply Corollary 2.2. Let N, X be matrices satisfying Assumption 3.5 and vector $y \in \mathbb{R}^d$ be the target output. Introducing the scalar $\lambda = x^\top Nx$, the pre-image equation becomes

$$(CN + \lambda I^d)x = y, \qquad \lambda = x^\top Nx. \tag{4}$$

In the following, we reduce the equation into a scalar polynomial equation of λ, and use the existence of λ to prove the existence of x. With $G = X^\top NX$, define

$$g(\lambda) = \det(G + \lambda I^m), \quad H(\lambda) = g(\lambda)I^d - X\,\mathrm{adj}(G + \lambda I^m)X^\top N, \tag{5}$$

where adj denotes the adjugate. For $\lambda g(\lambda) \neq 0$[6], the linear equation has the unique solution

$$x = (CN + \lambda I^d)^{-1}y = \frac{H(\lambda)y}{\lambda g(\lambda)}. \tag{6}$$

Substituting this expression into $\lambda = x^\top Nx$ and clearing the denominator gives

$$p_y(\lambda) := \lambda^3 g(\lambda)^2 - (H(\lambda)y)^\top N(H(\lambda)y) = 0. \tag{7}$$

The polynomial g is monic of degree m, and each entry of H has degree at most m, since the adjugate entries have degree at most $m - 1$. Thus the first term of p_y is monic of degree $2m + 3$, while the second has degree at most $2m$. By Corollary 2.2, this odd-degree univariate polynomial has a real root.

To make the above argument go through, we need to ensure that a root does not make the denominator $\lambda g(\lambda)$ zero. Under Assumption 3.5, Lemma 5 in the appendix proves that this cannot happen for almost every target y. For such a target, choose a real root λ^* and let $x^* = H(\lambda^*)y/(\lambda^* g(\lambda^*))$ (Equation (6)). Then $p_y(\lambda^*) = 0$ gives $x^{*\top}Nx^* = \lambda^*$, while the inverse formula gives $(CN + \lambda^* I^d)x^* = y$. Hence x^* is a root to $F_{N,X}(x) = y$. □

[5] For fixed N, the map from input to output prefixes is polynomial with equal input and output dimensions, so measure-zero sets of input prefixes have measure-zero images by Lemma 4.

[6] We will explain why this is true later in the proof sketch

The assumptions fail only on a measure-zero set. The next lemma shows that Assumption 3.5 holds almost everywhere.

Lemma 2 (Exceptional Parameters and Prefixes)**.** *Fix integers $d \geq 1$ and $m \geq 0$. The set of parameters violating Assumption 3.5 has measure zero in $\mathbb{R}^{d\times d}$ when $m = 0$, and in $\mathbb{R}^{d\times d} \times \mathbb{R}^{d\times m}$ when $m \geq 1$. Consequently, for each $m \geq 1$ and almost every fixed N, the set of matrices X for which Assumption 3.5 fails has measure zero in $\mathbb{R}^{d\times m}$.*

We reduce each failure to a polynomial equation in (N, X), such as $\det\big(X^\top NX\big) = 0$. Explicit examples show that none of these polynomials is identically zero. Their zero sets, and their finite union, therefore have measure zero by Lemma 3. We give the details in Section B.5.

Proof sketch of Theorem 3.4. Together, Lemmas 1 and 2 give Theorem 3.4 by induction on the sequence length. For almost every fixed N, the exceptional input prefixes have measure zero, and so do their output images by Lemma 4. Outside these images, every reachable output prefix has a pre-image satisfying Assumption 3.5, so almost every next output is reachable. Starting with the first output, Theorem A.1 completes the induction. The full induction and the transfer back to the original parameters (K, Q, V) are given in Section B.6. □

It is natural to ask whether the original attention mechanism is also surjective. We show that it is not through the following theorem.

Theorem 3.6. Attn *is not almost always surjective.*

We defer the proof of this theorem to the Appendix. Here we provide a brief overview of why attention itself is not surjective. For a positive-measure set of parameters, the symmetric part of $K^\top Q$ is negative definite. If the first output b_1 is kept in a bounded region, then a_1 is also bounded when V is invertible. At the second position, the self-attention weight on Va_2 decays like $\exp\big(a_2^\top K^\top Q a_2\big)$ as $\|a_2\|$ grows, while the cross term involving a_1 grows only linearly in $\|a_2\|$. Consequently b_2 remains bounded whenever b_1 is bounded, so output pairs with bounded b_1 and sufficiently large b_2 form a positive-measure unreachable set. We provide more such negative results in the Appendix.

4 Implications of Surjectivity on Modern Networks and Applications

In this section, we discuss how our theoretical results on the surjectivity of large models from Section 3 relate to safety concerns in real-world settings, particularly with respect to adversarial attacks. Our goal is not to provide an exhaustive list of all practical models or attacks. Instead, we present concrete examples of generative models used across diverse application areas — such as language, vision, and robotics — to illustrate the scope and implications of the surjectivity theory.

4.1 Language Models

By Theorems 3.1 and 3.2, Pre-LayerNorm is surjective; since it is used in every layer of modern Transformers, the Transformer architecture itself is also surjective.

Corollary 4.1. TF *is almost always surjective, and so are compositions of* TF.

Our results are proved in embedding space and assume direct control over the input embedding, unlike typical LLM deployments. Language models take discrete tokens mapped through a fixed embedding function, and most generative models are decoder-only and autoregressive. Thus our theorems do not directly apply to prompt-based attacks that exploit autoregressive generation (Zou et al., 2023). This limitation is arguably reassuring (Theorem C.3): if surjectivity held in that setting, it could imply a broader vulnerability surface.

However, our results raise questions about how to interpret language-model input-output behavior. Copyright concerns often cite a model's ability to reproduce specific outputs, such as a sentence from a proprietary source (He et al., 2025), as evidence. Our findings imply that, in principle, any sentence can be produced by decoding the final-layer output from a suitable input embedding, even if the model was trained autoregressively. In the Appendix, we demonstrate this on GPT-2 (Radford et al.,

2018): decoding the last layer yields a 37-word sentence from a 2025 New York Times article that could not have appeared in training. Thus, caution is needed when inferring private or copyrighted data from outputs alone, especially for inputs outside typical usage patterns.

4.2 VISION MODELS

Diffusion models are a class of generative models originally proposed for image generation (Sohl-Dickstein et al., 2015). They are now widely applied to other domains, including video generation (Liu et al., 2024), robotics (Chi et al., 2023), and beyond. Here we describe the generation process of diffusion models. Let the data we want to generate be represented by a vector $x \in \mathbb{R}^d$. First, one generates $x(0) \sim \mathcal{N}\left(0, I^d\right)$ from a Gaussian distribution. We regard x as a variable depending on $t \in [0, 1]$. After that, x evolves, or diffuses, according to a velocity vector field $v(x, t) \in \mathbb{R}^d$. The field v is trained in the hope that $x(1)$ follows the same distribution as the data distribution $p(x)$ that we want to generate. Formally we have

$$\mathrm{d}x/\mathrm{d}t = v(x, t), x(0) \sim \mathcal{N}\left(0, I^d\right), x(1) \sim p(x).$$

In practice directly solving this equation is often intractable, so we discretize $[0, 1]$ into intervals $\{[z_k, z_{k+1}]\}_{k\in[m-1]}$ with $z_1 = 0, z_m = 1$ and generate via approximation

$$x\left(z_{k+1}\right) = x(z_k) + v(x(z_k), z_k)(z_{k+1} - z_k), \text{for all } k \in [m-1]. \tag{8}$$

Early approaches to diffusion models inject Gaussian noise at inference time (see e.g. Ho et al., 2020), resulting in stochastic updates that led to high quality outputs but slowed down the generation process. Here, we focus on the more recent *deterministic* ODE solvers (Song et al., 2022) as described above, which also benefit from faster generation. Velocity predictor v is usually parameterized as a U-Net (Ronneberger et al., 2015; Ho et al., 2020) or Transformer (Peebles & Xie, 2023). When v is implemented using a transformer, x is tokenized into a sequence of vectors before being passed to the Transformer.

For image generation, a noisy image $x(0)$ evolves under v into a noiseless image $x(1)$. By Equation (8), since the first layer of v is typically a normalization (GroupNorm for U-Nets and Pre-LayerNorm for Transformers), Theorems 3.1 and 3.2 imply that deterministic diffusion models are almost always surjective from noise space to output space, suggesting inherent vulnerability to adversarial attacks. Indeed, prior work (Zeng et al., 2024) constructed noise vectors $x(0)$ that generate harmful content; our results show that, regardless of training, such an input exists for any output image.

4.3 ROBOTICS

Neural networks, in particular sequence models, are increasingly common in robotics, and are making robots increasingly powerful. However, this also gives rise to safety concerns given our results. As an example, let us take a policy network from prior work (Radosavovic et al., 2024), which follows a widely used design in practice. The network is implemented by a causally masked Transformer, i.e. compositions of TF. At timestep t, the action b_t[7] is generated by the Transformer on input sequence $a_1, b_1, \cdots, a_{t-1}, b_{t-1}, a_t$. Here a is the sequence of observations from the environment.

Note that this sequence model diverges from the ones we studied in the earlier section, by interleaving the true input sequence (observations) and previous outputs (actions). This interleaving is done to improve the smoothness of robot actions. Using similar techniques from Section 3, we can prove that this policy network is almost always surjective.

Theorem 4.2. *Let* Rob *be a composition of* TF *blocks. Given a sequence* a*, we iteratively calculate sequence* b *as* $b_t = \mathrm{Rob}(a_1, b_1, \cdots, b_{t-1}, a_t), t \geq 2; b_1 = \mathrm{Rob}(a_1)$*. This defines a function* f *from* a *to* b*.* f *is almost always surjective.*

We defer the proof to the Appendix. This theorem means that for the policy network described in Radosavovic et al. (2024), almost any action sequence can be induced by some corresponding sequence of inputs — such as a video clip played for the robot — regardless of how undesirable or unsafe the resulting behavior may be.

[7]Robotics convention uses a for action, but we use b here to keep notations in this paper consistent.

AI USE STATEMENT

We used generative AI tools to polish the writing and assist in drafting theorem statements and proofs. The main research ideas were developed by the human authors, who prompted AI tools to draft proofs and then revised and verified the resulting arguments. In particular, AI tools helped identify missing genericity assumptions in the proof of Theorem 3.4. The authors reviewed the AI-assisted material and take responsibility for the final content of this work, including all mathematical claims and proofs.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. PMLR, 2017.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c0c783b5fc0d7d808f1d14a6e9c8280d-Paper.pdf.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016. URL https://arxiv.org/abs/1409.0473.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, pp. 02783649241273668, 2023.

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

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/fe131d7f5a6b38b23cc967316c13dae2-Paper.pdf.

Aaron Grattafiori, Abhimanyu Dubey, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. Olmo: Accelerating the science of language models, 2024. URL https://arxiv.org/abs/2402.00838.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Luxi He, Yangsibo Huang, Weijia Shi, Tinghao Xie, Haotian Liu, Yue Wang, Luke Zettlemoyer, Chiyuan Zhang, Danqi Chen, and Peter Henderson. Fantastic copyrighted beasts and how (not) to generate them, 2025. URL https://arxiv.org/abs/2406.14526.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023. URL https://arxiv.org/abs/1606.08415.

Morris W. Hirsch. *Differential Topology*, volume 33 of *Graduate Texts in Mathematics*. Springer New York, NY, 1 edition, 1976. ISBN 978-0-387-90148-0. doi: 10.1007/978-1-4684-9449-5. URL https://doi.org/10.1007/978-1-4684-9449-5. Also available as eBook, ISBN 978-1-4684-9449-5; Softcover ISBN 978-1-4684-9451-8 (2012).

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1535–1546, 2017.

Imagen-Team-Google, Jason Baldridge, et al. Imagen 3, 2024. URL https://arxiv.org/abs/2408.07009.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: Llm-based input-output safeguard for human-ai conversations, 2023. URL https://arxiv.org/abs/2312.06674.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

Myeongseob Ko, Henry Li, Zhun Wang, Jonathan Patsenker, Jiachen T. Wang, Qinbin Li, Ming Jin, Dawn Song, and Ruoxi Jia. Boosting alignment for post-unlearning text-to-image generative models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 85131–85154. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/9aa51796f8bede2ea947d6b6e3087ab8-Paper-Conference.pdf.

Pengxiang Li, Lu Yin, and Shiwei Liu. Mix-ln: Unleashing the power of deeper layers by combining pre-ln and post-ln, 2024. URL https://arxiv.org/abs/2412.13795.

Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models, 2024. URL https://arxiv.org/abs/2402.17177.

James R. Munkres. *Elements of Algebraic Topology*. Addison-Wesley, Menlo Park, CA, 1984.

OpenAI, Josh Achiam, et al. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1314–1324, 2018.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Real-world humanoid locomotion with reinforcement learning. *Science Robotics*, 9(89):eadi9579, 2024.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241. Springer, 2015.

Tianneng Shi, Kaijie Zhu, Zhun Wang, Yuqi Jia, Will Cai, Weida Liang, Haonan Wang, Hend Alzahrani, Joshua Lu, Kenji Kawaguchi, et al. PromptArmor: Simple yet effective prompt injection defenses. *arXiv preprint arXiv:2507.15219*, 2025. URL https://arxiv.org/abs/2507.15219.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023. URL https://arxiv.org/abs/2307.08621.

Gemini Robotics Team, Saminda Abeyruwan, et al. Gemini robotics: Bringing ai into the physical world, 2025. URL https://arxiv.org/abs/2503.20020.

Hugo Touvron, Louis Martin, et al. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Zijun Wang, Haoqin Tu, Jieru Mei, Bingchen Zhao, Yisen Wang, and Cihang Xie. Attngcg: Enhancing jailbreaking attacks on llms with attention manipulation, 2024. URL https://arxiv.org/abs/2410.09040.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems*, 36:80079–80110, 2023.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

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

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training, 2024. URL https://arxiv.org/abs/2312.06635.

Yaopei Zeng, Yuanpu Cao, Bochuan Cao, Yurui Chang, Jinghui Chen, and Lu Lin. Advi2i: Adversarial image attack on image-to-image diffusion models, 2024. URL https://arxiv.org/abs/2410.21471.

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

Jiachen Zhu, Xinlei Chen, Kaiming He, Yann LeCun, and Zhuang Liu. Transformers without normalization. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14901–14911, 2025.

Zhijian Zhuo, Yutao Zeng, Ya Wang, Sijun Zhang, Jian Yang, Xiaoqing Li, Xun Zhou, and Jinwen Ma. Hybridnorm: Towards stable and efficient transformer training via hybrid normalization, 2025. URL https://arxiv.org/abs/2503.04598.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023. URL https://arxiv.org/abs/2307.15043.

A Measure-Theoretic Preliminaries

Here we present some basic results from measure theory. In this paper, measure means Lebesgue measure on the indicated Euclidean space. The following form of Fubini's theorem allows us to pass between a joint exceptional set and exceptional sets with some variables fixed.

Theorem A.1 (Fubini's Theorem for Measure-Zero Sets)**.** *Let $r, s \geq 1$ and let $E \subset \mathbb{R}^r \times \mathbb{R}^s$ be Lebesgue measurable. For almost every $u \in \mathbb{R}^r$, the section $E_u = \{v \in \mathbb{R}^s : (u, v) \in E\}$ is Lebesgue measurable. The set E has measure zero if and only if E_u has measure zero for almost every $u \in \mathbb{R}^r$.*

In particular, an exceptional set may depend on the fixed variables. We do not need to use the same exceptional set for every section.

Lemma 3 (Zero Sets of Polynomials)**.** *Let $r \geq 1$ and let $p : \mathbb{R}^r \to \mathbb{R}$ be a polynomial with real coefficients that is not identically zero. Then $\{u \in \mathbb{R}^r : p(u) = 0\}$ has measure zero.*

Proof. We use induction on r. A nonzero univariate polynomial has finitely many roots, so the statement holds for $r = 1$. For $r > 1$, write $u = (v, t) \in \mathbb{R}^{r-1} \times \mathbb{R}$ and

$$p(v, t) = \sum_{j=0}^{k} a_j(v) t^j,$$

where at least one coefficient polynomial, say a_{j_0}, is not identically zero. By induction, the set $\{v : a_{j_0}(v) = 0\}$ has measure zero. For every other v, the polynomial $t \mapsto p(v, t)$ is not identically zero and has finitely many roots. The zero set of p is closed and hence measurable. Applying Theorem A.1 to its sections proves the claim. □

Lemma 4 (Images of Measure-Zero Sets)**.** *Let $r \geq 1$, let $F : \mathbb{R}^r \to \mathbb{R}^r$ be continuously differentiable, and let $E \subset \mathbb{R}^r$ have measure zero. Then $F(E)$ has measure zero.*

Proof. Fix an integer $R \geq 1$ and put $D_R = [-R, R]^r$. Since DF is continuous, there is a finite $L \geq 1$ such that $\|DF(u)\| \leq L$ for all $u \in D_R$. Integrating the derivative along the line segment between $u, v \in D_R$ gives $\|F(u) - F(v)\| \leq L\|u - v\|$.

For any $\varepsilon > 0$, cover $E \cap D_R$ by countably many cubes of side lengths ℓ_j with $\sum_j \ell_j^r < \varepsilon$. In each cube that meets $E \cap D_R$, choose a point u_j in this intersection. Every other point u in the intersection satisfies $\|u - u_j\| \leq \sqrt{r}\, \ell_j$. Its image therefore lies in a cube centered at $F(u_j)$ of side length $2L\sqrt{r}\, \ell_j$. These image cubes cover $F(E \cap D_R)$ and have total volume at most $(2L\sqrt{r})^r \varepsilon$. Since ε is arbitrary, $F(E \cap D_R)$ has measure zero. Finally, $F(E) = \bigcup_{R=1}^{\infty} F(E \cap D_R)$ is a countable union of measure-zero sets. □

The equality of the input and output dimensions in Lemma 4 is important. We will apply it to polynomial maps between input and output prefixes of the same length.

B Omitted Proofs in Section 2, Section 3, and Section 4

B.1 Odd-Degree Univariate Polynomials

Proof of Corollary 2.2. Fix $y \in \mathbb{R}$ and write

$$q(x) = p(x) - y = c_n x^n + \sum_{k=0}^{n-1} c_k x^k,$$

where $n \geq 1$ is odd, $c_n \neq 0$, and all coefficients are real. Set $B = \sum_{k=0}^{n-1} |c_k|$ and choose $R > \max\{1, B/|c_n|\}$. For $x \in \{-R, R\}$, we have

$$|q(x) - c_n x^n| \leq \sum_{k=0}^{n-1} |c_k| R^k \leq B R^{n-1} < |c_n| R^n.$$

Thus $q(R)$ has the same sign as $c_n R^n$, while $q(-R)$ has the same sign as $c_n(-R)^n$. Since n is odd, these signs are opposite. The polynomial q is continuous on $[-R, R]$, so Theorem 2.1 gives an $x \in (-R, R)$ such that $q(x) = 0$, or equivalently $p(x) = y$. Since y was arbitrary, p is surjective onto $\mathbb{R}$. Taking $y = 0$ gives a real root of p. □

B.2 Pre-LayerNorm for Sequences

Theorem 3.2. *Let $f : \oplus_{i\in[n]}\mathbb{R}^d \to \oplus_{i\in[n]}\mathbb{R}^d$ be a continuous function, then $g : \oplus_{i\in[n]}\mathbb{R}^d \to \oplus_{i\in[n]}\mathbb{R}^d$ defined by $g(a)_i = f(\mathrm{LN}(a))_i + a_i$ is surjective.*

Proof. The proof is very similar to that of Theorem 3.1. Let

$$M = \sup_{a\in\oplus_{i\in[n]}\mathbb{R}^d} \|f(\mathrm{LN}(a))\| < \infty.$$

The finiteness of M follows because each $\mathrm{LN}(a_i)$ has 2-norm at most $\sqrt{d}\|\gamma\| + \|\beta\|$, so $\mathrm{LN}(a)$ lies in a compact ball in $\oplus_{i\in[n]}\mathbb{R}^d$ on which f is bounded. For any specific output sequence $b \in \oplus_{i\in[n]}\mathbb{R}^d$ for which we want to find a corresponding input, construct $R = M + \|b\| + 1$. Then applying Theorem 2.3 to the function $F(a) = b - f(\mathrm{LN}(a))$ restricted to $B^{nd}(R)$ proves the existence of a corresponding input. Hence g is surjective. □

Like Theorem 3.1, this proof can also be extended to other normalizations that are continuous and have bounded output.

B.3 Surjectivity for a Fixed Prefix

Proof of Lemma 1. Empty prefix, $m = 0$. Since $N + N^\top \neq 0$, the polynomial $q(y) = y^\top N y = \frac{1}{2}y^\top(N + N^\top)y$ is not identically zero. Indeed, if a symmetric matrix A satisfies $y^\top A y = 0$ for every y, testing $y = e_i$ and $y = e_i + e_j$ shows that every entry of A is zero. By Lemma 3, the set $\{y : q(y) = 0\}$ has measure zero. For every other y, let $t = \sqrt[3]{q(y)}$ be the real cube root and set $x = y/t$. Then

$$F_{N,X}(x) = (x^\top N x)x = \frac{q(y)}{t^3}y = y.$$

This proves the claim for $m = 0$.

Prefixes with $1 \leq m < d$. Fix a target $y \in \mathbb{R}^d$. Introducing the scalar $\lambda = x^\top N x$, we can write the pre-image equation as

$$(CN + \lambda I^d)x = y, \qquad \lambda = x^\top N x. \tag{9}$$

We first solve the linear equation for a fixed λ and then enforce the second equality. Define

$$\begin{aligned} g(\lambda) &= \det(G + \lambda I^m), \\ H(\lambda) &= g(\lambda)I^d - X\,\mathrm{adj}(G + \lambda I^m)X^\top N, \end{aligned} \tag{10}$$

where adj denotes the adjugate, which satisfies $A\,\mathrm{adj}(A) = \mathrm{adj}(A)A = \det(A)I$ for every square matrix A.

Suppose for now that $\lambda g(\lambda) \neq 0$. Put $B = G + \lambda I^m$, so B is invertible. Since $G = X^\top N X$, direct multiplication gives

$$\begin{aligned} &(\lambda I^d + XX^\top N)(I^d - XB^{-1}X^\top N) \\ &\quad = \lambda I^d + X[I^m - \lambda B^{-1} - GB^{-1}]X^\top N = \lambda I^d. \end{aligned}$$

Thus $CN + \lambda I^d$ is invertible, with

$$(CN + \lambda I^d)^{-1} = \frac{1}{\lambda}(I^d - XB^{-1}X^\top N) = \frac{H(\lambda)}{\lambda g(\lambda)}. \tag{11}$$

Substituting the resulting candidate $x = H(\lambda)y/(\lambda g(\lambda))$ into $\lambda = x^\top Nx$ gives

$$p_y(\lambda) := \lambda^3 g(\lambda)^2 - (H(\lambda)y)^\top N(H(\lambda)y) = 0. \tag{12}$$

The polynomial g is monic of degree m. Each entry of $\operatorname{adj}(G + \lambda I^m)$ has degree at most $m-1$, so each entry of H has degree at most m. Therefore the second term in p_y has degree at most $2m$, whereas the first term is monic of degree $2m+3$. Consequently, p_y is a real univariate polynomial of odd degree $2m+3$. By Corollary 2.2, which follows from the intermediate value theorem in Theorem 2.1, it has a real root.

Clearing the denominator could introduce a root at which Equation (11) is undefined. The supporting lemma below, Lemma 5, proves that there is a measure-zero set $E \subset \mathbb{R}^d$ such that, for every $y \notin E$, every real root λ_* of p_y satisfies $\lambda_* g(\lambda_*) \neq 0$. For such a target, choose a real root and set

$$x = \frac{H(\lambda_*)y}{\lambda_* g(\lambda_*)}.$$

By Equation (11), $(CN + \lambda_* I^d)x = y$. Moreover, dividing $p_y(\lambda_*) = 0$ by $\lambda_*^2 g(\lambda_*)^2$ gives $x^\top Nx = \lambda_*$. These two identities show that $F_{N,X}(x) = y$. Thus every target outside E has a pre-image.

Prefixes with $m \geq d$. Here C is positive definite and N is invertible. Fix any $y \in \mathbb{R}^d$ and define

$$T_y(x) = N^{-1}(C + xx^\top)^{-1}y.$$

Let $\mu > 0$ be the smallest eigenvalue of C. For every $u \in \mathbb{R}^d$,

$$u^\top(C + xx^\top)u = u^\top Cu + (x^\top u)^2 \geq \mu\|u\|^2.$$

Hence $C + xx^\top$ is invertible for every x, and its inverse has norm at most $1/\mu$. The map T_y is continuous and satisfies

$$\|T_y(x)\| \leq \frac{\|N^{-1}\|\,\|y\|}{\mu}.$$

Taking $R = \|N^{-1}\|\,\|y\|/\mu + 1$, we obtain a continuous map from $B^d(R)$ into itself. By Brouwer's fixed point theorem, Theorem 2.3, there is an $x \in B^d(R)$ with $T_y(x) = x$. Multiplying this equality by N and then by $C + xx^\top$ gives

$$y = (C + xx^\top)Nx = CNx + (x^\top Nx)x = F_{N,X}(x).$$

Since y was arbitrary, $F_{N,X}$ is surjective in this case. □

B.4 Nonzero Denominator

We now prove the supporting lemma used in the scalar reduction. It treats $\lambda = 0$ and the real roots of g separately.

Lemma 5 (Nonzero Denominator). *Let d, m be integers with $1 \leq m < d$, and let $N \in \mathbb{R}^{d\times d}$ and $X \in \mathbb{R}^{d\times m}$ satisfy Assumption 3.5. With $G = X^\top NX$, define*

$$\begin{aligned} g(\lambda) &= \det(G + \lambda I^m), \\ H(\lambda) &= g(\lambda)I^d - X\operatorname{adj}(G + \lambda I^m)X^\top N, \\ p_y(\lambda) &= \lambda^3 g(\lambda)^2 - (H(\lambda)y)^\top N(H(\lambda)y), \qquad y \in \mathbb{R}^d, \end{aligned}$$

where adj *is the adjugate. Then there is a measure-zero set $E \subset \mathbb{R}^d$ such that, for every $y \notin E$ and every real λ satisfying $p_y(\lambda) = 0$, we have $\lambda g(\lambda) \neq 0$.*

Proof. Excluding $\lambda = 0$. Define

$$P = I^d - XG^{-1}X^\top N.$$

We have $X^\top NP = X^\top N - GG^{-1}X^\top N = 0$, so the image of P lies in $\ker(X^\top N)$. Conversely, if $z \in \ker(X^\top N)$, then $Pz = z$. Therefore

$$\operatorname{Im} P = \ker(X^\top N).$$

Since G is invertible, $g(0) = \det G \neq 0$ and $\mathrm{adj}(G) = g(0)G^{-1}$. It follows that

$$H(0) = g(0)P, \qquad p_y(0) = -g(0)^2 (Py)^\top N(Py).$$

By assumption, there is a $z \in \ker(X^\top N)$ with $z^\top N z \neq 0$. Evaluating at $y = z$ and using $Pz = z$ shows that $y \mapsto (Py)^\top N(Py)$ is a nonzero polynomial. Thus

$$E_0 = \{y \in \mathbb{R}^d : (Py)^\top N(Py) = 0\}$$

has measure zero by Lemma 3. For $y \notin E_0$, the number 0 is not a root of p_y.

Excluding a real root of g. Fix a real α with $g(\alpha) = 0$ and put $A = G + \alpha I^m$. Since G is invertible, $\alpha \neq 0$. The identity $g(\lambda) = (-1)^m \det(-\lambda I^m - G)$ shows that the roots of g are the negatives of the roots of the characteristic polynomial of G, with the same multiplicities. Thus α is a simple root of g and $g'(\alpha) \neq 0$. Differentiating the determinant by its diagonal entries gives

$$g'(\alpha) = \mathrm{tr}(\mathrm{adj}(A)).$$

In particular, $\mathrm{adj}(A) \neq 0$. For $m \geq 2$, its entries are signed $(m-1) \times (m-1)$ minors, so A has rank at least $m - 1$. Since $\det A = 0$, its rank is exactly $m - 1$. For $m = 1$, $A = [0]$ has rank 0 and adjugate $[1]$, giving the same conclusion.

Let $v, w \in \mathbb{R}^m$ be nonzero vectors spanning the right and left kernels of A, respectively. The identities $A\,\mathrm{adj}(A) = 0$ and $\mathrm{adj}(A)A = 0$ show that every column of the adjugate is a multiple of v and every row is a multiple of $w^\top$. Since the adjugate is nonzero, there is a real $\kappa \neq 0$ such that

$$\mathrm{adj}(A) = \kappa v w^\top.$$

Consequently,

$$H(\alpha)y = -\kappa X v (w^\top X^\top N y),$$
$$p_y(\alpha) = -\kappa^2 (Xv)^\top N(Xv)(w^\top X^\top N y)^2.$$

Both factors needed here are nonzero. First, $Av = 0$ gives $Gv = -\alpha v$, and hence

$$(Xv)^\top N(Xv) = v^\top G v = -\alpha \|v\|^2 \neq 0.$$

Second, the row vector $w^\top X^\top N$ cannot be zero: otherwise multiplication by X would give $w^\top G = 0$, whereas $w^\top A = 0$ gives $w^\top G = -\alpha w^\top \neq 0$. Thus

$$E_\alpha = \{y \in \mathbb{R}^d : w^\top X^\top N y = 0\}$$

is the zero set of a nonzero linear polynomial. It has measure zero by Lemma 3, and $p_y(\alpha) \neq 0$ whenever $y \notin E_\alpha$.

There are finitely many real roots of g. Therefore

$$E = E_0 \cup \bigcup_{\substack{\alpha \in \mathbb{R} \\ g(\alpha) = 0}} E_\alpha$$

has measure zero. If $y \notin E$, neither 0 nor any real root of g is a root of p_y. These are precisely the real values at which $\lambda g(\lambda) = 0$, proving the claim. □

B.5 Verification of the Retention Conditions

Proof of Lemma 2. We express each failure of Assumption 3.5 through the vanishing of a nonzero polynomial, then apply Lemma 3.

The condition for $m = 0$. The equality $N + N^\top = 0$ is equivalent to

$$\sum_{i,j=1}^{d} (N_{ij} + N_{ji})^2 = 0.$$

This is a polynomial in the entries of N, and it is nonzero at $N = I^d$. Its zero set therefore has measure zero.

The conditions for $1 \le m < d$. Since $G = X^\top N X$, its entries are polynomials in the entries of (N, X). The failure of invertibility is exactly $\det G = 0$.

To detect repeated roots, let $\chi_G(t) = \det(tI^m - G)$ and let $\Delta(G)$ be its discriminant. If $\rho_1, \dots, \rho_m$ are the complex roots counted with multiplicity, then

$$\Delta(G) = \prod_{1 \le i < j \le m} (\rho_i - \rho_j)^2.$$

For $m = 1$, this empty product is 1. The discriminant is a polynomial in the coefficients of the monic polynomial χ_G, hence in the entries of G and of (N, X). It vanishes exactly when two roots coincide. Thus the second failure is exactly $\Delta(G) = 0$.

For the condition on the kernel, define the polynomial matrix

$$Z = \det(G) I^d - X \operatorname{adj}(G) X^\top N$$

and the polynomial

$$\Psi(N, X) = \sum_{i,j=1}^{d} \left[Z^\top (N + N^\top) Z \right]_{ij}^2.$$

If G is invertible, the projection $P = I^d - X G^{-1} X^\top N$ used in Lemma 5 satisfies $Z = \det(G) P$ and $\operatorname{Im} Z = \ker(X^\top N)$. Therefore $z^\top N z = 0$ for every $z \in \ker(X^\top N)$ if and only if

$$u^\top Z^\top (N + N^\top) Z u = 0 \qquad \text{for every } u \in \mathbb{R}^d.$$

The matrix in this quadratic form is symmetric. As in the $m = 0$ part of the proof of Lemma 1, testing coordinate vectors and their pairwise sums shows that its quadratic form is identically zero if and only if the matrix itself is zero. This is equivalent to $\Psi(N, X) = 0$. Once the singular case $\det G = 0$ is included, every failure is therefore contained in

$$\{\det G = 0\} \cup \{\Delta(G) = 0\} \cup \{\Psi(N, X) = 0\}. \tag{13}$$

It remains to verify that none of these three polynomials is identically zero on the structured variables. Choose distinct positive numbers $c_1, \dots, c_m$ and set

$$N = I^d, \qquad X = [\sqrt{c_1} e_1, \dots, \sqrt{c_m} e_m].$$

Then $G = \operatorname{diag}(c_1, \dots, c_m)$ is invertible and has distinct eigenvalues, so $\det G \neq 0$ and $\Delta(G) \neq 0$. Also

$$P = \operatorname{diag}(\underbrace{0, \dots, 0}_{m}, \underbrace{1, \dots, 1}_{d-m}), \qquad Z = \det(G) P.$$

It follows that $Z^\top (N + N^\top) Z = 2 \det(G)^2 P$, and hence

$$\Psi(N, X) = 4 \det(G)^4 (d - m) > 0.$$

Each set in Equation (13) has measure zero by Lemma 3, so their union has measure zero.

The conditions for $m \ge d$. The matrix $C = X X^\top$ is symmetric positive semidefinite because $u^\top C u = \|X^\top u\|^2 \ge 0$. It fails to be positive definite exactly when $\det(X X^\top) = 0$. The failure of invertibility of N is $\det N = 0$. Both determinants are polynomials in (N, X), and both are nonzero at

$$N = I^d, \qquad X = [I^d, 0],$$

where the zero block has $m - d$ columns. Thus each failure set has measure zero by Lemma 3.

These descriptions also show that the exceptional sets are measurable: they are specified by polynomial equalities and their complements. Applying Theorem A.1 to the joint exceptional set in (N, X) shows that, for almost every fixed N, its section in X has measure zero. This proves the final assertion for every $m \ge 1$. □

B.6 From One Output to an Entire Sequence

Proof of Theorem 3.4. Fix d and n. By Lemma 2, outside a measure-zero set of matrices N, the following statements hold simultaneously: $N + N^\top \neq 0$, and, for every $m = 1, \dots, n-1$, the set $\mathcal{B}_m$ of input prefixes $X \in \mathbb{R}^{d\times m}$ violating Assumption 3.5 has measure zero. There are only finitely many prefix lengths, so the union of their exceptional sets of N still has measure zero. Fix such an N.

For each $m \geq 1$, define the polynomial map

$$T_m : (\mathbb{R}^d)^m \to (\mathbb{R}^d)^m,$$

$$T_m(x_1, \dots, x_m)_j = \sum_{i=1}^{j} (x_i^\top N x_j) x_i, \qquad j = 1, \dots, m. \tag{14}$$

We identify an input tuple $(x_1, \dots, x_m)$ with its matrix of columns X. We will prove by induction that T_m reaches almost every output prefix for $m = 1, \dots, n$.

For $m = 1$, the map is $T_1(x_1) = (x_1^\top N x_1)x_1$. Applying Lemma 1 with an empty prefix shows that its image has a measure-zero complement.

Now suppose $1 \leq m < n$ and the complement of $T_m((\mathbb{R}^d)^m)$ has measure zero. The input set $\mathcal{B}_m$ has measure zero by the choice of N. Since T_m is a polynomial map between spaces of the same dimension dm, Lemma 4 implies that $T_m(\mathcal{B}_m)$ has measure zero as well. Consequently,

$$U_m = \big((\mathbb{R}^d)^m \setminus T_m((\mathbb{R}^d)^m)\big) \cup T_m(\mathcal{B}_m)$$

has measure zero.

Fix an output prefix $u \notin U_m$. Since u lies in the image of T_m, it has an input prefix X with $T_m(X) = u$. This X cannot belong to $\mathcal{B}_m$, since otherwise $u \in T_m(\mathcal{B}_m)$. Hence Assumption 3.5 holds for (N, X). By Lemma 1, for almost every next output $y \in \mathbb{R}^d$, there is an $x \in \mathbb{R}^d$ with $F_{N,X}(x) = y$. The causal form in Equation (14) then gives

$$T_{m+1}(X, x) = (u, y).$$

Thus, for every $u \notin U_m$, the set of unreachable extensions y has measure zero. The exceptional set of y may depend on the chosen pre-image X. We only need one such pre-image for each fixed u; no common exceptional set across different prefixes is required.

To apply Theorem A.1, we check measurability. For any $k \geq 1$, continuity gives

$$T_k(\mathbb{R}^{dk}) = \bigcup_{R=1}^{\infty} T_k([-R, R]^{dk}).$$

Each set on the right is compact. Therefore the image is a countable union of closed sets, and its complement is Borel measurable. Applying Theorem A.1 to

$$(\mathbb{R}^d)^{m+1} \setminus T_{m+1}((\mathbb{R}^d)^{m+1})$$

and its sections over $u \in (\mathbb{R}^d)^m$ proves that this complement has measure zero. This completes the induction.

Finally, we transfer the conclusion from N to the original parameters. The sets $\{K : \det K = 0\}$ and $\{V : \det V = 0\}$ have measure zero by Lemma 3; their defining determinants are nonzero at I^d. Fix invertible K, V. The change of parameters

$$Q \longmapsto N = V^{-\top} K^\top Q V^{-1}$$

is invertible and linear, with inverse

$$N \longmapsto Q = K^{-\top} V^\top N V.$$

By Lemma 4, the exceptional set of N therefore corresponds to a measure-zero set of Q for every fixed invertible pair (K, V). To justify the joint statement, we may enlarge the exceptional set of N to a Borel set of measure zero. Its pre-image under the continuous map $(K, Q, V) \mapsto V^{-\top} K^\top Q V^{-1}$ on the open set of invertible K, V is measurable. Applying Theorem A.1 over (K, V) and Q, and adding the measure-zero sets of singular K or V, gives a measure-zero exceptional set of parameter triples.

For every remaining triple, $a_i \mapsto x_i = V a_i$ is a bijection on each input token, and Equations (2) and (14) show that $\mathrm{Ret}(a) = T_n(x)$. Thus Ret and T_n have the same output image. The image of T_n has a measure-zero complement by the induction, which proves the theorem. □

B.7 SOFTMAX ATTENTION

Theorem 3.6. Attn *is not almost always surjective.*

Proof. It suffices to consider sequences of length two. We exhibit a positive-measure set of parameters for which a positive-measure set of output pairs cannot be reached. Let $A = K^\top Q$, and consider parameters for which V is invertible and the symmetric part $(A + A^\top)/2$ is negative definite. This set has positive measure because it contains the product of an open neighborhood of $(K, Q) = (I^d, -I^d)$ with the open set of invertible V.

Fix such parameters. There exists $c > 0$ such that

$$u^\top A u \le -c\|u\|^2, \quad \text{for all } u \in \mathbb{R}^d.$$

Fix $B > 0$. Suppose an output pair (b_1, b_2) with $\|b_1\| \le B$ has a pre-image (a_1, a_2). Since the first attention output is $b_1 = Va_1$, we have

$$\|a_1\| \le \|V^{-1}\| B.$$

For the second output,

$$b_2 = \frac{\exp\big(a_1^\top A a_2\big) V a_1 + \exp\big(a_2^\top A a_2\big) V a_2}{\exp\big(a_1^\top A a_2\big) + \exp\big(a_2^\top A a_2\big)}.$$

Write $r = a_1^\top A a_2$, $q = a_2^\top A a_2$, and

$$\lambda = \frac{\exp(q)}{\exp(r) + \exp(q)}.$$

Then $b_2 = (1 - \lambda) b_1 + \lambda V a_2$. Since

$$q \le -c\|a_2\|^2, \qquad |r| \le \|A\|\|V^{-1}\| B \|a_2\|,$$

we have

$$\lambda \|V a_2\| \le \|V\|\|a_2\| \exp\left(-c\|a_2\|^2 + \|A\|\|V^{-1}\| B \|a_2\|\right).$$

The right-hand side is bounded uniformly over a_2; call this bound M_B. Therefore every reachable output pair with $\|b_1\| \le B$ must satisfy

$$\|b_2\| \le B + M_B.$$

Hence the positive-measure set

$$\left\{(b_1, b_2) \in \mathbb{R}^d \oplus \mathbb{R}^d \middle| \|b_1\| < B,\ B + M_B + 1 < \|b_2\| < B + M_B + 2\right\}$$

is disjoint from the image of the length-two attention map. For longer sequences, taking the Cartesian product with any positive-measure set of remaining outputs gives the same obstruction. Thus attention is not surjective, even up to a zero-measure exceptional subset of outputs, on a positive-measure set of parameters. Therefore Attn is not almost always surjective. □

B.8 ROBOTICS

Theorem 4.2. *Let* Rob *be a composition of* TF *blocks. Given a sequence* a*, we iteratively calculate sequence* b *as* $b_t = \mathrm{Rob}(a_1, b_1, \cdots, b_{t-1}, a_t), t \ge 2; b_1 = \mathrm{Rob}(a_1)$*. This defines a function* f *from* a *to* b*.* f *is almost always surjective.*

Proof. The proof is by induction and resembles the proof of Theorem 3.4. Output b_1 only depends on a_1. By Corollary 4.1 we know that $b_1 = \mathrm{Rob}(a_1) = \mathrm{TF}(a_1)$ is surjective. When constructing a_j, we assume that all $a_1, \cdots, a_{j-1}$ have already been determined by $b_1, \cdots, b_{j-1}$. In this way, b_j only depends on a_j, and the dependence is through a function that is a composition of functions with Pre-LayerNorms. By Theorem 3.1 we know that this function is surjective. In conclusion, Rob is surjective. □

C Additional Results

C.1 Negative Results on Surjectivity

Theorem C.1. *Let $d_1 > d$ and consider the class of two-layer ReLU MLPs $f : \mathbb{R}^d \to \mathbb{R}^d$ of the form*

$$f(x) = W_2\mathrm{ReLU}(W_1x + \lambda_1) + \lambda_2,$$

where $W_1 \in \mathbb{R}^{d_1\times d}$, $W_2 \in \mathbb{R}^{d\times d_1}$, $\lambda_1 \in \mathbb{R}^{d_1}$, and $\lambda_2 \in \mathbb{R}^d$. This class is not almost always surjective.

Proof. We exhibit a positive measure set of parameters for which f is not surjective. Let

$$\Omega = \{(W_1, W_2, \lambda_1, \lambda_2) \mid (W_2)_{1j} > 0 \text{ for every } j \in [d_1]\}.$$

This set has positive Lebesgue measure in the parameter space. For any parameters in Ω and any $x \in \mathbb{R}^d$, every coordinate of $\mathrm{ReLU}(W_1x + \lambda_1)$ is nonnegative. Therefore the first coordinate of the output satisfies

$$f(x)_1 = \sum_{j=1}^{d_1}(W_2)_{1j}\mathrm{ReLU}(W_1x + \lambda_1)_j + (\lambda_2)_1 \geq (\lambda_2)_1.$$

Hence no vector $y \in \mathbb{R}^d$ with $y_1 < (\lambda_2)_1$ belongs to the image of f. Thus every network with parameters in Ω is not surjective. Since Ω has positive measure, the non-surjective parameter set is not measure zero, and the class is not almost always surjective. □

We can prove a similar negative result for MLPs with GeLU activation.

Theorem C.2. *Let $d_1 > d$ and consider the class of two-layer GeLU MLPs $f : \mathbb{R}^d \to \mathbb{R}^d$ of the form*

$$f(x) = W_2\mathrm{GeLU}(W_1x + \lambda_1) + \lambda_2,$$

where $W_1 \in \mathbb{R}^{d_1\times d}$, $W_2 \in \mathbb{R}^{d\times d_1}$, $\lambda_1 \in \mathbb{R}^{d_1}$, and $\lambda_2 \in \mathbb{R}^d$. This class is not almost always surjective.

Proof. Let

$$m_{\mathrm{GeLU}} = \inf_{t\in\mathbb{R}} \mathrm{GeLU}(t).$$

This quantity is finite because scalar GeLU is continuous, tends to 0 as $t \to -\infty$, and tends to ∞ as $t \to \infty$.

As in the ReLU case, consider the positive measure set of parameters

$$\Omega = \{(W_1, W_2, \lambda_1, \lambda_2) \mid (W_2)_{1j} > 0 \text{ for every } j \in [d_1]\}.$$

For any parameters in Ω and any $x \in \mathbb{R}^d$, each coordinate of $\mathrm{GeLU}(W_1x + \lambda_1)$ is at least m_{GeLU}. Therefore

$$f(x)_1 = \sum_{j=1}^{d_1}(W_2)_{1j}\mathrm{GeLU}(W_1x + \lambda_1)_j + (\lambda_2)_1 \geq m_{\mathrm{GeLU}}\sum_{j=1}^{d_1}(W_2)_{1j} + (\lambda_2)_1.$$

Thus the first coordinate of $f(x)$ is uniformly lower bounded over all $x \in \mathbb{R}^d$. Any vector $y \in \mathbb{R}^d$ with

$$y_1 < m_{\mathrm{GeLU}}\sum_{j=1}^{d_1}(W_2)_{1j} + (\lambda_2)_1$$

is not in the image of f. Hence every network with parameters in Ω is not surjective. Since Ω has positive measure, the class is not almost always surjective. □

Theorem C.3. *Let $\mathcal{T}_\theta$ be any finite composition of* TF. *For any prompt length $n \in \mathbb{N}^+$, define the two-step autoregressive map*

$$\mathrm{AR}_{\theta,n} : \oplus_{i\in[n]}\mathbb{R}^d \to \mathbb{R}^d \oplus \mathbb{R}^d$$

as follows. Given input sequence $a = (a_1, \cdots, a_n)$, let

$$b_1 = \mathcal{T}_\theta(a_1, \cdots, a_n)_n, \qquad b_2 = \mathcal{T}_\theta(a_1, \cdots, a_n, b_1)_{n+1},$$

and set $\mathrm{AR}_{\theta,n}(a) = (b_1, b_2)$. The class of maps $\mathrm{AR}_{\theta,n}$ is not almost always surjective. In fact, no such $\mathrm{AR}_{\theta,n}$ is surjective.

Proof. The central observation is that the residual connection passes the input at each position to the output at the same position, while the non-residual terms are evaluated after LayerNorm and are therefore bounded. Recall from Section 2 that one Transformer block takes the form

$$\mathrm{TF}(x)_i = x_i + \mathrm{Attn}(\mathrm{LN}(x))_i + W_2\mathrm{GeLU}\left(W_1\mathrm{LN}\left(x_i + \mathrm{Attn}(\mathrm{LN}(x))_i\right) + \lambda_1\right) + \lambda_2.$$

Hence the only part of $\mathrm{TF}(x)_i$ that can grow directly with x_i is the residual term x_i.

We now bound the two remaining terms. Since LayerNorm has bounded image, there exists $B_{\mathrm{LN}} < \infty$ such that

$$\|\mathrm{LN}(z)\| \le B_{\mathrm{LN}}, \quad \text{for all } z \in \mathbb{R}^d.$$

For the attention branch,

$$\mathrm{Attn}(\mathrm{LN}(x))_i = \frac{1}{Z_i}\sum_{j=1}^{i} \exp\left(\mathrm{LN}(x_j)^\top K^\top Q\mathrm{LN}(x_i)\right) V\mathrm{LN}(x_j),$$

where the coefficients $\exp(\cdot)/Z_i$ are nonnegative and sum to one. Thus $\mathrm{Attn}(\mathrm{LN}(x))_i$ is a convex combination of vectors $V\mathrm{LN}(x_j)$, and

$$\|\mathrm{Attn}(\mathrm{LN}(x))_i\| \le \|V\|B_{\mathrm{LN}}.$$

In particular, if

$$c_i = x_i + \mathrm{Attn}(\mathrm{LN}(x))_i,$$

then $c_i - x_i$ is uniformly bounded, independently of the sequence length and the input.

For the MLP branch, the input is $\mathrm{LN}(c_i)$, which again lies in the bounded image of LayerNorm. Therefore $W_1\mathrm{LN}(c_i) + \lambda_1$ lies in a bounded subset of $\mathbb{R}^{d'}$. Since scalar GeLU is bounded on bounded intervals, the map

$$u \mapsto W_2\mathrm{GeLU}(W_1u + \lambda_1) + \lambda_2$$

is bounded on $\mathrm{LN}(\mathbb{R}^d)$. Hence there exists $B_{\mathrm{MLP}} < \infty$ such that

$$\|W_2\mathrm{GeLU}(W_1\mathrm{LN}(c_i) + \lambda_1) + \lambda_2\| \le B_{\mathrm{MLP}}.$$

Combining the attention and MLP bounds, there exists a constant

$$C_{\mathrm{block}} = \|V\|B_{\mathrm{LN}} + B_{\mathrm{MLP}} < \infty$$

such that for every sequence length m, every input sequence $x = (x_1, \cdots, x_m)$, and every position $i \in [m]$,

$$\|\mathrm{TF}(x)_i - x_i\| \le C_{\mathrm{block}}. \tag{15}$$

Since $\mathcal{T}_\theta$ is a finite composition of such blocks, summing the bounds over blocks gives a constant $C_\theta < \infty$ such that, for every sequence length m, every sequence x, and every position i,

$$\|\mathcal{T}_\theta(x)_i - x_i\| \le C_\theta. \tag{16}$$

Now take any prompt $a = (a_1, \cdots, a_n)$ and let $\mathrm{AR}_{\theta,n}(a) = (b_1, b_2)$. In the second autoregressive step, the input at position $n+1$ is exactly b_1. Applying Equation (16) to the sequence $(a_1, \cdots, a_n, b_1)$ at position $n+1$ gives

$$\|b_2 - b_1\| = \|\mathcal{T}_\theta(a_1, \cdots, a_n, b_1)_{n+1} - b_1\| \le C_\theta.$$

Thus the image of $\mathrm{AR}_{\theta,n}$ is contained in

$$\left\{(y_1, y_2) \in \mathbb{R}^d \oplus \mathbb{R}^d \mid \|y_2 - y_1\| \le C_\theta\right\}.$$

This is a proper subset of $\mathbb{R}^d \oplus \mathbb{R}^d$: for example, any pair $(0, y)$ with $\|y\| > C_\theta$ is not in the image. Hence $\mathrm{AR}_{\theta,n}$ is not surjective. Since this holds for every choice of finite parameters θ, the class is not almost always surjective. □

D BROADER DISCUSSION ON THE IMPLICATIONS OF SURJECTIVITY.

In this paper, we introduced the study of surjectivity of neural networks as a concrete formalization of studying the power of safety training and jailbreak vulnerabilities. In this section, we dive into the broader discussion of what surjectivity is, what it is not, and what surjectivity of modern networks implies for AI safety research.

D.1 THEORETICAL IMPLICATIONS AND THE FORMALISM OF SURJECTIVITY

On Theoretical Implications of Surjectivity. One major goal of safety training is to limit a model's ability to generate harmful outcomes. From a theoretical perspective, surjectivity implies that a model is vulnerable to jailbreaks in principle. That is, every outcome including those that are considered harmful by the model providers, can be generated by some input.

We make no claim in the other direction. In particular, it is possible that many or even every harmful behavior by a non-surjective model can still be elicited by some input, while the model's lack of surjectivity is due to its inability to produce other non-harmful behavior.

On Surjectivity versus Model Capability. Model capability perhaps is best captured through details of the input-output behavior. By design, our study of surjectivity is agnostic to whether the input-output relationship a model captures is a complex or an interesting one. Indeed, even simple functions, such as the *identity* function, can be surjective, while they are often not capturing interesting input-output behaviors. Our study of surjectivity intentionally decouples risks that are rooted in an attacker's ability to elicit particular behaviors — which is the main consideration of jailbreaks — from domain-specific risks that arise from having highly capable AI models in certain areas (such as bioweapons, etc.) in the first place.

One can take for granted that models that have undergone safety-training (and as a result are the main subject of the study of jailbreaks) are already highly capable models that have been trained on troves of relevant data to capture the complex relationship between the input and output spaces that deviates significantly from simple functions such as the identity. By establishing that these trained models are almost always surjective, our work highlights their inherent vulnerability, regardless of how capable the models are. That is, with enough information and computational power, an attacker can elicit any behavior from the model, including harmful ones.

On Surjectivity Versus Having Full Support. One might ask how surjectivity is different from a common assumption that generative models have full support? This question stems from viewing the outcome of the neural network as a stochastic function from input to the output space, while in this work we view fully trained networks as deterministic functions. When considering fully trained generative models as deterministic functions, we find two perspectives to be instructive.

The first perspective is to consider generative models with deterministic decoding, e.g., decoding the probability distribution greedily, with beam search, or with temperatures close to 0. In this perspective, the observation that a stochastic function has full support does not imply that its deterministic decoding schemes are surjective.

A second perspective is to consider the hidden embedding computed by the network as a deterministic function. Take GPT, for example, as a function that computes vector b, the hidden embedding output of its last block. Embedding b is used by the model to output any token i from a finite set of tokens with embeddings h_i, with probability $p(i \mid b) \propto \exp\left(b^\top h_i\right)$. GPT has full support, i.e., $p(i \mid b) > 0$ for all i, for any finite hidden embedding b. On the other hand, our surjectivity results can make a particular token arbitrarily likely when its embedding has a direction that separates it from the other token embeddings. Take an arbitrary token i with this property. Surjectivity of the network implies that for some input, the transformer's hidden embedding is $b = \lambda h_i$ for large λ. If $h_i^\top h_i > h_i^\top h_j$ for all other token embeddings h_j, then $p(j|b) \to 0$ for all $h_j \neq h_i$ as $\lambda \to \infty$. Thus surjectivity of the transformer implies that such a token can be made the deterministic greedy-decoding output.

On Surjectivity versus Impact of Implicit Regularization. There is a line of work (Du et al., 2018; Arora et al., 2019) suggesting that standard optimization algorithms such as gradient descent provide implicit regularization, encouraging the models to bias towards lower-rank parameters. A natural

question is whether such structure would impact our analysis on being almost always surjective. We would like to remark that the "low-rank" phenomena typically correspond to approximately low-rank matrices whose singular values decay but are not exactly zero in practice. These matrices are still full rank, and hence the corresponding linear maps have range equal to the full output space. From the perspective of our results, such models remain in the generic region where surjectivity holds; the main effect of regularization is to make certain output directions correspond to small singular directions, so that reaching them may require large-norm or highly atypical inputs. This is precisely the regime in which adversarial optimization can search for off-distribution inputs that exploit these directions, which is consistent with the "in-principle" vulnerability captured by surjectivity.

D.2 Practical Implications of Surjectivity in Safety

On computational and statistical considerations. In practice, the relationship between surjectivity and jailbreak vulnerabilities is more nuanced than explored in Section D.1. Surjectivity is an existential rather than a constructive statement about the mapping between input and output spaces: the existence of a pre-image x for a harmful output y does not imply that such a pre-image can be found computationally or information-theoretically efficiently.

From a computational perspective, finding such a pre-image may be intractable in the worst case. However, worst-case hardness results do not provide meaningful safety guarantees, since attackers need only succeed on some harmful instances and may have access to significant computational resources. The proliferation of jailbreaks in practice suggests that computational difficulty is rarely a bottleneck in the average or practical case.

From an information-theoretic perspective, surjectivity-based risks depend on the attacker's knowledge of the target outcome y. For example, in the language domain, a model's surjectivity demonstrates moderate risk, as it could point to *repeated-after-me* attacks. Though this attack would let the model output harmful content, it requires that the attacker knows the harmful text they want to elicit in detail. They cannot elicit sensitive hidden information that is a priori unavailable to the attacker (e.g., personally identifiable information, bioweapon risks). In other domains, such as robotics, the risk is more severe. For example, when outcome y represents an action in the physical domain — such as the trajectory of an autonomous drone or a robot arm — the knowledge of the exact trajectory a drone must take to hit an object at a destructive speed may be known to the attacker. In these settings, surjectivity presents a significant concern as an attacker with knowledge of y can craft an input that elicits the destructive behavior.

Still, many jailbreaks that do uncover new information can be framed in terms of surjectivity. For example, *suffix-injection* attacks (Wei et al., 2023; Zou et al., 2023; Wang et al., 2024) work by eliciting outputs of the form $y = ab$, where a is a fixed prefix (e.g., "Certainly! Here is ...", "Sure, the answer is ...") and b contains novel potentially harmful information the model providers have sought to prohibit. Here the attacker's goal is not to learn anything from a itself, but to ensure that the model commits to a as the starting condition, exploiting the autoregressive continuation that makes generating b more likely. In other words, even though a carries no new information, the ability to force such a prefix (as captured through the lens of surjectivity) significantly increases the probability of eliciting harmful b.

More broadly, we believe that a natural direction for future work is to examine how much partial information about y is sufficient to reconstruct a problematic input x — for instance, quantifying what fraction of the output tokens must be fixed or how many adaptive queries an attacker can make before finding an x that produces an output in the vicinity of y.

On Implications of Surjectivity on Safety Interventions. Surjectivity also bears on the discourse in the AI safety community, in particular about two types of overall practical approaches to safety, which we summarize as "train-for-safety" (Grattafiori et al., 2024; Ko et al., 2024) versus "filter-for-safety" (Inan et al., 2023; Shi et al., 2025). The former includes works on post-training for safety, RLHF/RLAIF, harmless finetuning, and safety pre-training that aim to bake restrictions on the output space into model weights at training time, while the latter includes task-specific filters and constitutional classifiers that post-hoc filter the generated outcomes. The train-for-safety methods offer low-latency and efficient generation policies that are highly desirable to the model providers compared to post-hoc filtering that discards harmful generated responses. The surjectivity of a model

can be taken as further evidence that the train-for-safety paradigm is not a sufficient line of defense on its own.

E EXPERIMENT

Our results prove surjectivity of practical architectures by proving surjectivity of their building blocks. Hence our proof also provides algorithms to find an input corresponding to a specific output of the network that we want. For GPT-2, or more generally surjective autoregressive models, we can find the input one token at a time as described in Algorithm 1.

Algorithm 1 Finding Input Sequence

Input: A Frozen Transformer TF, An Output Sequence b
Output: A Reconstructed Sequence a

```
a_1 ← 0
Optimize a_1 using gradient descent on loss (b_1 − TF(a_1))^2
for i = 2 to n do
    a_i ← 0
    Optimize a_i using gradient descent on loss (b_i − TF(a_1, ··· , a_i)_i)^2
end for
return a
```

In this section, we conduct experiments to verify the theoretical statements using GPT-2. In particular we implement Algorithm 1 to find inputs corresponding to the following outputs:

- Twenty sentences from 2025 New York Times articles. GPT-2 could not have been trained on such sentences. One example is “The United States and China said Monday they reached an agreement ... threatening the world’s two largest economies”. The length of these sentences varies from 7 words to 37 words.
- Twenty-five sentences that contain completely random words from vocabulary, with length 2, 4, ..., 50. One example is "whims produ ether debunked depressive FoundingeeshedonApplication Weight refin 58".

Notice that Algorithm 1 cannot guarantee that we always find a corresponding a because gradient-based optimization is still a heuristic algorithm. However, decoding one input at a time is a lot simpler than jointly optimizing the whole sequence a at once. If this optimization is still too difficult, we can further decompose the algorithm into finding the hidden embeddings iteratively. However we find that for GPT-2 Algorithm 1 is enough.

For every gradient descent, we set learning rate to be 0.1 and optimize for 200 steps. We use an A100 GPU for inference. For all forty-five sentences we described above, the algorithm succeeds in finding the corresponding input sequences. The decoding speed per token is 10.25 ± 0.14 seconds.

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: The abstract and introduction state the paper's scope: surjectivity results for neural-network building blocks, the topological tools used to prove them, and safety implications. The formal statements and experiments in the main text and appendix support these claims.

   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: The broader discussion and safety sections discuss limitations of surjectivity as an existential, non-constructive statement, including computational and information-theoretic caveats.

   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: The main theoretical results state their assumptions explicitly, and proofs are provided either in the main text or appendix. The preliminary section states the topological tools used in the proofs and cites the relevant references.

   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: The appendix describes the reconstruction experiment, model choice, optimization algorithm, learning rate, number of steps, and compute used. Code to reproduce the experiments will be provided in the supplementary material.

   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: [Yes]

   Justification: Code and reproduction instructions will be included in the supplementary material. The experiments use publicly available model assets and prompts described in the appendix.

   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: The appendix specifies the experimental protocol, including the optimization procedure, learning rate, number of optimization steps, model, and output sequences used for evaluation.

   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: The appendix reports success over all tested examples and reports decoding speed with variability. The experiments are intended as reproducibility demonstrations rather than statistical comparisons between methods.

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: The appendix reports the GPU type used for inference and the per-token decoding time. Additional execution details will be included with the supplementary code.

   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: The work is theoretical with a small reproducibility experiment using public model assets, and it discusses potential safety implications and misuse considerations. We are not aware of any deviation from the NeurIPS Code of Ethics.

   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: [Yes]

    Justification: The safety section and broader discussion analyze how surjectivity relates to jailbreak vulnerability, model safety, and possible misuse, while also explaining how the results can clarify limitations of safety interventions.

    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 paper does not release new high-risk models or scraped datasets. The supplementary code is for reproducing the reported experiments with existing public model assets.

    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: The paper cites the existing model architectures and prior works used in the discussion and experiments. The supplementary material will document code dependencies and public model assets used for reproduction.

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: [Yes]

Justification: The only new asset released with the submission is reproduction code, which will be documented in the supplementary material. The paper does not introduce a new dataset or trained model.

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: The paper does not involve crowdsourcing, user studies, or 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: The paper does not involve crowdsourcing, user studies, or research with human subjects, so IRB approval is not applicable.

    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: [Yes]

    Justification: The AI use statement discloses assistance with writing, drafting theorem statements and proofs, and identifying missing genericity assumptions in the proof of Theorem 3.4. The human authors revised and verified the AI-assisted arguments and take responsibility for the final content.

    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.