MDPI

Article

Dimension-independent Approximations on Low-dimensional Manifolds using Transformers

Ji Shi [1] **and Demetrio Labate** [2,*]

[1] Department of Applied Mathematics, University of Houston; shijibest@gmail.com
[2] Department of Applied Mathematics, University of Houston; dlabate@uh.edu
* Correspondence: dlabate@uh.edu

Abstract

Deep neural networks have been remarkably successful in high-dimensional learning and scientific computing, often succeeding where classical discretization methods fail due to the curse of dimensionality. This efficacy is often explained by their approximation properties combined with the manifold hypothesis: the idea that although data are embedded in dimension D, the effective degrees of freedom are governed by a much smaller intrinsic dimension $d \ll D$. Under this hypothesis, data are concentrated near a low-dimensional manifold that neural networks can approximate efficiently. While the approximation theory for fully-connected ReLU networks on manifolds is well established, a comparable theory for transformer architectures, the dominant model class in modern foundation models, is still emerging. In this paper, we prove a new *non-asymptotic*, uniform approximation theorem for a class of single-head ReLU-transformers acting on vector inputs, where the approximation error depends only on the intrinsic dimension d rather than on the ambient dimension D. To the best of our knowledge, this is the first transformer approximation result that combines an intrinsic-dimensional rate with an ambient-dimension-independent multiplicative constant. We include a numerical experiment using a circle embedded in ambient dimensions of various sizes, showing that the observed error remains nearly unchanged as D varies, in agreement with the predicted ambient-dimension independence.

Keywords: Approximation theory; deep neural networks; expressive power; Hölder continuity; manifold learning; transformer networks.)

1. Introduction

Transformers have evolved from a sequence-processing architecture into a general-purpose template for learning maps between high-dimensional objects. Large language models, vision transformers, and diffusion transformers all share the same primitive: a self-attention operator that mixes token representations through data-dependent interactions [1,2]. From the perspective of approximation theory, self-attention can be viewed as a structured multilinear algorithm that enables *global communication* across tokens and *programmable gating* of information flow. This naturally raises the following fundamental mathematical question.

When the ambient input dimension is high, can a transformer approximate a smooth target function with complexity governed by the intrinsic *degrees of freedom of the data rather than by the nominal dimension?*

We address this question through a rigorous approximation-theoretic framework motivated by the manifold hypothesis, which posits that real-world high-dimensional

Received:
Revised:
Accepted:
Published:

data sets often lie on low-dimensional latent manifolds embedded in the high-dimensional space [3]. More formally, let $\mathcal{M} \subset [0,1]^D$ be a smooth compact d-dimensional Riemannian submanifold with $d \ll D$, and let $f : \mathcal{M} \to \mathbb{R}$ belong to a Hölder class $\mathcal{H}(\beta, \mathcal{M}, M)$ with $\beta \in (0,1]$. Our goal is a uniform approximation bound on $\mathcal{M}$ by a transformer network T (Definition 2) whose architectural hyperparameters are explicit and whose error constants do not deteriorate with D. The transformer model we analyze is a single-head ReLU-attention architecture (Definitions 3–4) acting on a tokenized embedding of a vector input. While practical transformers typically use softmax attention, ReLU-gated attention is a standard theoretical surrogate: it preserves the key feature needed for our construction—data-dependent routing—and it allows exact piecewise-linear algebraic identities inside the network.

Our main result (Theorem 1) shows that *transformers approximate Hölder functions on manifolds at an intrinsic-dimensional rate with an ambient-dimension-independent constant*. That is, for every resolution parameter $N \in \mathbb{N}$, given a Hölder function f with values in a $d-$dimensional manifold embedded in $\mathbb{R}^D$, there exists a transformer T with fixed-size feed-forward sublayers such that

$$\sup_{\mathbf{x}\in\mathcal{M}} |T(\mathbf{x}) - f(\mathbf{x})| \;\leq\; C(\beta, d_e, M, \mathrm{diam}(\mathcal{M}))\, N^{-\beta/d_e}, \tag{1}$$

where d_e is an *effective dimension* depending only on the manifold dimension d and the geometric regularity of $\mathcal{M}$, and, crucially, the multiplicative constant C does *not* depend on the ambient dimension D. This is the transformer analogue of the "dimension-independent constant" phenomenon previously obtained for fully-connected ReLU networks on manifolds [4]. We complement this theorem with a numerical experiment showing that, for the same circle embedded in ambient dimensions $D = 2, 4, \ldots, 512$, the observed approximation error remains nearly unchanged as D varies.

1.1. Why constant independence matters

For approximations on a d-dimensional manifold, it is understood that an approximation rate of the form $N^{-\beta/d}$ reflects the intrinsic dimension. However, for a high ambient dimension D, a bound can still be vacuous if the multiplicative approximation constant scales poorly in D. Controlling this constant is especially important for applications in which D can be extremely large (e.g., high-resolution images, scientific states, or token embeddings). Remarkably, in our estimate (1), the ambient dimension enters only through a single linear map in the embedding layer; after this one-time "compression" step, the transformer blocks operate in a regime whose quantitative complexity is governed by d and the manifold geometry only.

1.2. Literature review

The approximation of functions on low-dimensional manifolds by deep *fully-connected* ReLU networks has a substantial literature. Early work [5] have established uniform approximation of smooth functions on smooth d-dimensional manifolds with rates governed by d. Subsequent results refined smoothness assumptions, network architectures, and statistical consequences [6–10]. A recurring technical theme is how to pass from a function on $\mathcal{M} \subset \mathbb{R}^D$ to a function on a low-dimensional Euclidean domain. Some works do this via explicit coordinate charts, i.e., by local parameterizations [6–9]; other works use a non-constructive approach based on global dimension reduction arguments [4,10–12]. In particular, Johnson–Lindenstrauss (JL)-type embeddings for manifolds [13] underpin a "global" reduction approach that is well suited to controlling constants, as we have shon in prior work [4]. Our approach in this paper follows this global dimension reduction strategy

but replaces the fully-connected approximator by an explicitly constructed transformer architecture.

Approximation theory has been developed for a number of neural architectures. For example, ResNet-type convolutional networks have been analyzed from both approximation and nonparametric estimation perspectives [14,15]. Due to their relevance Neural Operators for the approximation of operators associated with PDEs, several studies have also investigated approximation and error bounds [16], including the recent development of a new class of Neural Operators explicitly designed to learn the mapping between functions defined on any Riemannian manifolds [17]. These works show that architectural structure (convolution, skip connections) can be leveraged to match statistical and approximation limits on certain low-dimensional or regularity-adapted function classes. They provide an important conceptual precedent: once an architecture becomes dominant in applications, it is natural to ask whether it also admits sharp approximation guarantees comparable to those available for fully-connected networks. Our paper takes an analogous step for transformers in a manifold-based setting.

Due to the increasing importance of transformers in current deep learning applications, there is a rapidly growing body of studies that focus on their expressivity and statistical behavior. Universal approximation results for sparse and dense transformers are established by [18–20], and a number of works derive quantitative approximation guarantees in settings tailored to transformers [21–24]. Recent papers also highlight algorithmic and statistical phenomena that appear transformer-specific, including efficient parity computation via chain-of-thought reasoning [25] and minimax-optimal nonparametric in-context learning [26]. Finally, [27] connects approximation/statistical theory to empirical scaling laws for transformers when the data are intrinsically low-dimensional. Despite these advances, existing results do not provide what is needed for our geometric question: a uniform manifold approximation theorem for transformers with an approximation constant independent of D together with an explicit non-asymptotic construction. Filling this gap is the main purpose of the present work, which establishes a uniform approximation bound for transformers on manifolds with a constant that does not depend on D, in direct analogy with [4] for fully-connected networks.

1.3. Relationship Between ReLU and Softmax Attention

Practical transformers typically use softmax attention [1]. In the notation of this paper, if

$$S = (KH)^\top QH \in \mathbb{R}^{l\times l},$$

then the corresponding self-attention layer can be written as

$$SA_{\text{softmax}}(H) = VH\text{Softmax}(S),$$

where softmax is applied columnwise to S. By contrast, the transformer model studied in this paper uses the ReLU attention mechanism

$$SA_{\text{ReLU}}(H) = VH\rho((KH)^\top QH),$$

where $\rho(x) = \max\{0, x\}$ is applied elementwise. Thus, the two mechanisms use the same bilinear score matrix $S = (KH)^\top QH$ and the difference lies in the nonlinear map applied to these scores.

More explicitly, for the ith query token and the jth key token, softmax attention assigns the weight

$$\alpha_{ji}^{\text{soft}}(S) = \frac{e^{S_{ji}}}{\sum_{m=1}^{l} e^{S_{mi}}},$$

whereas ReLU attention assigns the weight

$$\alpha_{ji}^{\mathrm{ReLU}}(S) = \rho(S_{ji}) = \max\{0, S_{ji}\}.$$

Hence, softmax attention produces normalized strictly positive weights,

$$\alpha_{ji}^{\mathrm{soft}}(S) > 0, \qquad \sum_{j=1}^{l} \alpha_{ji}^{\mathrm{soft}}(S) = 1,$$

while ReLU attention has the exact zeroing property

$$S_{ji} < 0 \quad \Longrightarrow \quad \alpha_{ji}^{\mathrm{ReLU}}(S) = 0.$$

This is the key structural difference for our proof. Softmax gives smooth globally coupled weighting through the normalization denominator, whereas ReLU gives piecewise-linear exact gating. In particular, if the matrices Q, K, and V are designed so that desired interactions correspond to positive scores and undesired interactions correspond to negative scores, then ReLU attention implements an exact sparse routing mask at finite scale.

This viewpoint is also consistent with recent theoretical transformer works that adopt ReLU-based attention in place of standard softmax attention, including [27–31].

There is also a useful asymptotic relation between the two mechanisms. If $s \in \mathbb{R}^l$ has a unique maximizer j^*, then

$$\mathrm{Softmax}(\tau s) \to e_{j^*} \qquad \text{as } \tau \to \infty,$$

so softmax approaches hard selection only asymptotically as the temperature is sharpened. Moreover, Hu et al. [32] show that softmax attention can approximate a generalized version of ReLU to arbitrary precision, while Lai et al. [30] show that ReLU-activated attention is a cubic spline and that replacing ReLU by a smooth activation such as SoftMax yields a smoothed counterpart of the original transformer. These results support the view that ReLU attention is a mathematically tractable surrogate that preserves the same score-based token-interaction mechanism while replacing smooth normalized weighting by exact piecewise-linear gating.

For this reason, the present paper uses ReLU attention as a constructive theoretical surrogate for softmax attention. The proof in Section 3.1 relies on exact zeroing and explicit arithmetic identities inside the attention block, and these are available under ReLU gating but not under standard softmax normalization. Therefore, our theorem should be interpreted as an approximation result for an analytically tractable attention variant that captures the same query–key interaction mechanism as practical softmax attention.

1.4. Our contributions

We emphasize the following main contributions.

- *First intrinsic-dimension transformer approximation on manifolds with a D-free constant.* We prove a uniform approximation bound for transformers on $\mathcal{M} \subset [0,1]^D$ in which the rate depends on an effective dimension d_e and the multiplicative constant is independent of the ambient dimension D; see Theorem 1 and Remark 2. To our knowledge, no previous transformer approximation result achieves this type of dimension-independent approximation estimate in the manifold setting. The closest analogue in the literature is the fully-connected ReLU result of [4], and our theorem can be viewed as a transformer counterpart of that phenomenon.

- *Non-asymptotic approximation with explicit architecture choices.* Our theorem is not merely asymptotic: for any prescribed $N \in \mathbb{N}$, we specify explicit network hyperparameters (number of blocks, number of tokens, embedding dimension, and FFN depth/width) and obtain a finite-N bound. In particular, the token embedding dimension is fixed at $d_{\text{embd}} = 5$ and the FFN sublayers have constant depth and width. This contrasts with existing approximation statements for attention models that are stated only in limiting regimes or in big-O form. The non-asymptotic nature of our bound aligns with contemporary questions about how approximation error scales with model size and complements scaling-law analyses such as [27].
- *A constructive transformer network for approximating Hölder target functions.* We introduce a novel and explicit Sparse Attention Interaction Gadget: by hand-designing the matrices Q, K, V (together with a constant-size tokenwise FFN), a single-head ReLU self-attention block can be forced to realize prescribed sparse pairwise token interaction including exact bilinear products and gated routing and we then compose these gadgets block-by-block to compile a classical partition-of-unity grid approximator for Hölder functions inside a transformer. Beyond yielding the error bound, this construction provides a reusable template for using attention blocks to implement arithmetic operations and gating primitives in approximation problems, in the spirit of "algorithmic" transformer analyses such as [25], but here targeted to a classical approximation-theoretic pipeline.
- *Numerical validation of the ambient-dimension-independent behavior.* We complement the theorem with a simple dimension-sweep experiment on the same circle embedded in $\mathbb{R}^D$ for $D = 2, 4, \ldots, 512$ and observe that the numerically approximated L^∞ error remains nearly unchanged as D varies; see Section 4.

The rest of the paper is organized as follows. Section 2 introduces notation, Hölder spaces and the transformer class. Section 3 states the main theorem and situates it among related results. Section 3.1 provides the proof and the explicit construction. Section 4 presents the numerical experiment.

2. Notation and Definitions

Throughout the paper, we denote by $\mathbb{N} = \{1, 2, \cdots\}$ the set of natural numbers and by $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$ the set of nonnegative integers. The floor of $a \in \mathbb{R}$, denoted $\lfloor a \rfloor$, is the greatest integer less than or equal to a, and the ceiling of a, denoted $\lceil a \rceil$, is the lowest integer greater than or equal to a. We use boldface symbols to denote vectors and regular fonts with a subscript to denote vector coordinates, e.g., $\mathbf{x} \in \mathbb{R}^n$ and x_i is the i-th coordinate of $\mathbf{x}$. For $\mathbf{x} \in \mathbb{R}^n$, we write $\|\mathbf{x}\|_2 = \left(\sum_{i=1}^n |x_i|^2\right)^{1/2}$ and $\|\mathbf{x}\|_\infty = \max_{1 \le i \le n} |x_i|$. For a matrix A, we write $A^\top$ for its transpose. For a multi-index $\alpha = (\alpha_1, \ldots, \alpha_D) \in \mathbb{N}_0^D$, we set $\|\alpha\|_1 := \sum_{i=1}^D \alpha_i$ and write $\partial^\alpha f := \partial_1^{\alpha_1} \cdots \partial_D^{\alpha_D} f$ for the corresponding mixed partial derivative. If $\mathcal{M}$ is a Riemannian manifold, $\Delta(\mathbf{x}, \mathbf{y})$ denotes the geodesic distance between $\mathbf{x}, \mathbf{y} \in \mathcal{M}$ and $\operatorname{diam}(\mathcal{M}) := \sup_{\mathbf{x},\mathbf{y} \in \mathcal{M}} \Delta(\mathbf{x}, \mathbf{y})$ its diameter.

We recall the definition of Hölder spaces on which we our approximation results.

Definition 1 (Hölder space). *Let $\beta > 0$ and $F \subseteq \mathbb{R}^D$ be a closed set. The Hölder space with degree of smoothness β on F is defined as*

$$\mathcal{H}(\beta, F) = \left\{ f \in C^{\lfloor \beta \rfloor}(F) \;\middle|\; ||f||_{\mathcal{H}(\beta,F)} < \infty \right\},$$

where, for $f : F \to \mathbb{R}$, the Hölder norm of f is defined by

$$||f||_{\mathcal{H}(\beta,F)} = \max\left\{ \max_{\alpha:\, ||\alpha||_1 \le \lfloor\beta\rfloor} \sup_{\mathbf{x}\in F} |\partial^\alpha f(\mathbf{x})|, \max_{\alpha:\, ||\alpha||_1 = \lfloor\beta\rfloor} \sup_{\substack{\mathbf{x},\mathbf{y}\in F \\ \mathbf{x}\neq\mathbf{y}}} \frac{|\partial^\alpha f(\mathbf{x}) - \partial^\alpha f(\mathbf{y})|}{||\mathbf{x}-\mathbf{y}||_2^{\beta-\lfloor\beta\rfloor}} \right\}.$$

For $M > 0$, we denote the closed ball in $\mathcal{H}(\beta, F)$ with radius M as

$$\mathcal{H}(\beta, F, M) := \left\{ f \in C^{\lfloor\beta\rfloor}(F) \mid ||f||_{\mathcal{H}(\beta,F)} \le M \right\}.$$

We adopt the following definition of a transformer neural network (TNN) and its realizations, in the spirit of approximation-theoretic treatments of network classes (cf. [4,23,27]).

Definition 2 (Transformer Neural Network). *We define a transformer neural network T as a composition of functions of the form*

$$T(\cdot) = \mathcal{D} \circ B_{L_T} \circ \cdots \circ B_1 \circ (E(\cdot) + P), \tag{2}$$

where

- *L_T: The number of transformer blocks B_i in T;*
- *E is a linear embedding layer: $\mathbb{R}^D \to \mathbb{R}^{d_{embd} \times l}$ defined as follows:*

$$E(\mathbf{x}) \;=\; r\,(U\mathbf{x})^\top S, \tag{3}$$

  where
  - *D is the input dimension;*
  - *d_{embd} is the token embedding dimension;*
  - *l is the number of hidden tokens;*
  - *$U \in \mathbb{R}^{m \times D}$ is a learnable weight matrix;*
  - *$r = (1, 0, \ldots, 0)^\top \in \mathbb{R}^{d_{embd}}$;*
  - *$S = \left[\begin{array}{ccccc|c|ccccc} 1 & 0 & 0 & \cdots & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{array}\right] \in \mathbb{R}^{m \times l}$, where l is a multiple of m;*

- *P is a trainable positional encoding matrix;*
- *$\mathcal{D}$ is a decoding layer: $\mathbb{R}^{d_{embd} \times l} \to \mathbb{R}$, which is fixed to compute the sum of the first row.*

Definition 2 requires the following additional definitions.

Definition 3 (Transformer block). *We define a transformer block B as a residual composition of form*

$$B(H) = FFN(SA(H) + H) + SA(H) + H,$$

where $H \in \mathbb{R}^{d_{embd} \times l}$ is an input as an embedding matrix, SA is a self-attention layer and FFN is a feed-forward layer. In Figure 1, we give a schematic of the transformer block.

Definition 4 (Self-attention layer). *We define a (single-head) self-attention as follows:*

$$SA_{Q,K,V}(H) = VH\varrho((KH)^\top QH),$$

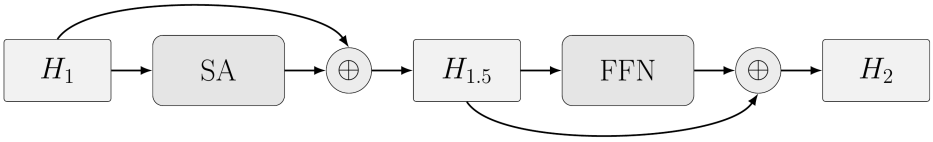

Figure 1. Schematic of a transformer block with single-head self-attention (SA) and a feed-forward sublayer (FFN).

where Q, K and $V \in \mathbb{R}^{d_{embd} \times d_{embd}}$ are the query, key, and value matrices and $\varrho(x) = \max\{0, x\}$ is applied elementwise. We will omit the dependence of SA on Q, K and V and write $SA(H)$ when no ambiguity arises.

Remark 1 (Single-head vs. Multi-head Attention)**.** *While practical transformer architectures routinely employ multi-head attention to improve optimization dynamics and capture diverse feature representations simultaneously [1], our theoretical analysis considers a single-head self-attention module. From the perspective of approximation theory, estimating a uniform error bound using a single-head architecture constitutes a mathematically stronger fundamental result. Specifically, any approximation capability achieved by a single-head transformer can be realized by a multi-head transformer of comparable size. Therefore, our construction demonstrates that the intrinsic ability of transformer networks to bypass the ambient dimension relies fundamentally on the core routing and gating mechanisms of self-attention itself, rather than the multiplicity of attention heads.*

Definition 5 (Feed-forward layer)**.** *A feed-forward layer of width W and depth L is the form*

$$FFN(h) = A_L\, \varrho(A_{L-1} \dots \varrho(A_1 h + b_1) \dots + b_{L-1}) + b_L,$$

where $h \in \mathbb{R}^{d_{\text{embd}}}$ is a token vector, $A_1 \in \mathbb{R}^{W \times d_{\text{embd}}}$, $A_2, \dots, A_{L-1} \in \mathbb{R}^{W \times W}$, $A_L \in \mathbb{R}^{d_{\text{embd}} \times W}$, and $b_1, \dots, b_{L-1} \in \mathbb{R}^W$, $b_L \in \mathbb{R}^{d_{\text{embd}}}$. Each feed-forward layer is applied tokenwise to an embedding matrix H.

Definition 6 (Feed-forward network class)**.**

$$\mathcal{FFN}(W, L) = \{FFN \mid FFN \text{ is a feed-forward network with at most } L \text{ layers with width } W\}.$$

Definition 7 (Transformer block class)**.**

$$\mathcal{B}(W, L) = \{B \mid B \text{ is a transformer block with a single headed self-attention layer and a } L \text{ deep feed-forward layer with width } W\}.$$

Definition 8 (Transformer network class)**.** *We define a class of transformer networks as follows:*

$$\mathcal{T}(L_T, l, d_{\text{embd}}, L_{FFN}, W_{FFN}) = \{T \mid T \text{ is defined in (2) with at most } L_T \text{ transformer blocks, token dimension } d_{\text{embd}} \text{ and } l \text{ hidden tokens, where each block contains a feed-forward network with at most } L_{FFN} \text{ layers and width at most } W_{FFN}\}.$$

3. Results

Our study considers functions in $\mathcal{H}(\beta, \mathcal{M}, M)$, where the domain $\mathcal{M} \subset [0,1]^D$ is a smooth compact d-dimensional manifold with $d < D$. Our main theorem proves that these

functions can be approximated using a transformer network whose parameters, except for the input embedding layer, depend on an intrinsic dimension d_e, which depends on d and the geometry of the manifold, but not on the ambient dimension D.

As indicated in the Introduction, we adopt a non-constructive approximation approach and apply a manifold version of the Johnson–Lindenstrauss lemma, a variant of a result by Eftekhari and Wakin [13], which establishes the existence of a linear transformation mapping points in $\mathcal{M} \subseteq [0,1]^D$ nearly isometrically into a lower dimensional domain.

The application of this nearly isometric mapping requires technical assumptions about the manifold. Namely, we assume that $\mathcal{M}$ is a compact d-dimensional Riemannian submanifold of $\mathbb{R}^D$ with a bounded condition number. We recall that the *condition number* of a manifold $\mathcal{M}$ is defined as $1/\tau$, where τ, called the *reach* of the manifold, is the largest number such that any point at distance less than τ from $\mathcal{M}$ has a unique nearest point on $\mathcal{M}$. Reach controls both local curvature (bounded by $1/\tau$) and global self-avoidance (cf. [33]). We also recall the diameter of a Riemannian manifold $\mathcal{M}$:

$$\operatorname{diam}(\mathcal{M}) := \sup_{\mathbf{x},\mathbf{y}\in\mathcal{M}} \Delta(\mathbf{x},\mathbf{y}),$$

where $\Delta(\mathbf{x},\mathbf{y})$ is the geodesic distance on $\mathcal{M}$.

We can now state our main approximation theorem, whose proof is given in Section 3.1and whose ambient-dimension-independent behavior is illustrated numerically in Section 4.

Theorem 1. *For $D \in \mathbb{N}$, let $\mathcal{M}$ be a compact d-dimensional Riemannian submanifold contained in $[0,1]^D$, with $\mathbf{0} \in \mathcal{M}$, having condition number $1/\tau$ and volume $V_{\mathcal{M}}$ satisfying $\frac{V_{\mathcal{M}}}{\tau^d} \geq \left(\frac{21}{2\sqrt{d}}\right)^d$, and let $f \in \mathcal{H}(\beta,\mathcal{M},M)$, where $M > 0$ and $\beta \in (0,1]$. Let*

$$d_e = \left\lceil 828\left(24d + 2d\log\left(\tfrac{9\sqrt{d}}{\tau}\right) + \log(2V_{\mathcal{M}}^2)\right)\right\rceil. \tag{4}$$

Then, for any $K \in \mathbb{N}$, there exists a transformer network

$$T \in \mathcal{T}(L_T, l, d_{embd}, L_{FFN}, W_{FFN})$$

with $L_T = (d_e+1)N + 1$, $l = d_e N$, $d_{embd} = 5$, $L_{FFN} = 3$ and $W_{FFN} = 5$ so that

$$\max_{\mathbf{x}\in\mathcal{M}} |T(\mathbf{x}) - f(\mathbf{x})| \leq 3\left(2^{d_e+3} d^{\beta} M\, diam(\mathcal{M})\right) N^{-\frac{\beta}{d_e}}. \tag{5}$$

Remark 2. *Theorem 1 provides a* fully explicit, non-asymptotic *uniform approximation estimate. In particular, for every $N \in \mathbb{N}$, the theorem constructs a transformer network T (with the explicit architecture scaling specified there) satisfying (5). Equivalently, there exists a constant $C = C(\beta, d_e, M, diam(\mathcal{M}))$, independent of D, such that*

$$\max_{\mathbf{x}\in\mathcal{M}} |T(\mathbf{x}) - f(\mathbf{x})| \leq C\, N^{-\frac{\beta}{d_e}}.$$

We stress that this estimate is non-asymptotic*: the bound holds for each finite N and comes with an explicit quantitative dependence on the network design parameters (depth/tokens), rather than being stated only as an asymptotic rate as $N \to \infty$. Our contribution establishes this transformer approximation estimate in the manifold setting with a multiplicative constant independent of D. To the best of our knowledge, this is the first transformer approximation result of this type that simultaneously avoids any dependence of the multiplicative constant on the ambient dimension D and provides an explicit finite N (non-asymptotic) uniform error bound.*

Remark 3. *The analogous ambient-dimension-independent approximation estimate for fully-connected ReLU networks established in Theorem 1 and Remark 5 of [4] states that, for $f \in H(\beta, M, \mathcal{M})$, there exists a ReLU network Φ_f such that*

$$\sup_{x\in M}\left|R(\Phi^f)(x) - f(x)\right| \le C_{\mathrm{FNN}}(\beta, d_e, \mathcal{M}, \mathrm{diam}(M))\left(N^2L^2\log^3(N+2)\right)^{-\beta/d_e},$$

where C_{FNN} is independent of the ambient dimension D.

By contrast, Theorem 1 yields a transformer T satisfying

$$\sup_{x\in M}|T(x) - f(x)| \le C_{\mathrm{Tr}}(\beta, d_e, \mathcal{M}, \mathrm{diam}(M))\, N^{-\beta/d_e}, \qquad L_T = (d_e+1)N+1, \qquad l = d_eN.$$

Hence, both results share the same geometric hallmark: the approximation rate is governed by the same effective dimension d_e, and the multiplicative constant is independent of the ambient dimension D. The main difference lies in the complexity parameterization. The fully-connected ReLU bound is expressed in terms of width and depth, whereas the present transformer bound is expressed in terms of the number of blocks and tokens in an explicit attention construction.

In particular, recall from Definition 2 that l denotes the number of hidden tokens in the transformer architecture, and Theorem 1 specifies

$$L_T = (d_e+1)N+1 \text{ and } l = d_eN.$$

Hence

$$N = \frac{L_T - 1}{d_e+1} = \frac{l}{d_e},$$

so the estimate in Theorem 1,

$$\sup_{x\in M}|T(x) - f(x)| \le C_{\mathrm{Tr}}\, N^{-\beta/d_e},$$

may equivalently be rewritten either in terms of the number L_T of transformer blocks or in terms of the number l of hidden tokens as

$$\sup_{x\in M}|T(x) - f(x)| \le \widetilde{C}_{\mathrm{Tr}}(L_T - 1)^{-\beta/d_e}$$

and

$$\sup_{x\in M}|T(x) - f(x)| \le \widehat{C}_{\mathrm{Tr}}\, l^{-\beta/d_e},$$

for suitable constants $\widetilde{C}_{\mathrm{Tr}}, \widehat{C}_{\mathrm{Tr}}$ depending on $\beta, d_e, \mathcal{M}, \mathrm{diam}(M)$, but still independent of the ambient dimension D. Therefore, the present theorem should be viewed not as improving the known ReLU-network rate, but rather as a transformer counterpart of the same ambient-dimension-independent constant phenomenon previously established for fully-connected ReLU networks.

Example 1 (How the bound can guide the network design)**.** *Theorem 1 can be used as a practical scaling rule once a baseline transformer has been trained. Indeed, if the observed error behaves like*

$$e_N \approx C_{\mathrm{Tr}}N^{-\beta/d_e},$$

then a baseline run with resolution parameter N_0 and validation error e_{N_0} yields the estimate

$$\widehat{C}_{\mathrm{Tr}} := e_{N_0}N_0^{\beta/d_e}.$$

For a target accuracy ε, the theorem then suggests choosing

$$N_\varepsilon = \left\lceil \left(\frac{\widehat{C}_{\mathrm{Tr}}}{\varepsilon} \right)^{d_e/\beta} \right\rceil,$$

which in turn gives the explicit architecture sizes

$$L_T = (d_e + 1)N_\varepsilon + 1, \qquad l = d_e N_\varepsilon.$$

For example, if $\beta = 1$, $d_e = 2$, a baseline model with $N_0 = 10$ gives $e_{N_0} = 0.16$, then

$$\widehat{C}_{\mathrm{Tr}} = 0.16 \cdot 10^{1/2} \approx 0.506.$$

To reach $\varepsilon = 0.10$, one may choose

$$N_\varepsilon = \left\lceil \left(\frac{0.506}{0.10} \right)^2 \right\rceil = 26,$$

hence

$$L_T = 79, \qquad l = 52.$$

This gives a concrete way to turn the approximation bound into a design rule. Section 4 complements this interpretation empirically by showing that, when the same ReLU-transformer architecture is tested on the same circle embedded in different ambient dimensions, the observed error remains nearly unchanged as D varies.

3.1. Proof of Theorem 1

Before proving Theorem 1, we introduce two useful lemmas.

Lemma 1 (Theorem 4 in [4]). *Let $\mathcal{M}$ be a compact K-dimensional Riemannian submanifold of $\mathbb{R}^D$ having condition number $1/\tau$ and volume $V_{\mathcal{M}}$, satisfying $\frac{V_{\mathcal{M}}}{\tau^K} \geq \left(\frac{21}{2\sqrt{K}}\right)^K$. Fix $\epsilon \in (0, 1/3]$ and $\rho \in (0, 1)$. Let A be a random $d_\epsilon \times D$ matrix populated with i.i.d. random variables $a_{i,j}$ where*

$$a_{ij} = \begin{cases} +\frac{1}{\sqrt{d_\epsilon}} \text{ with probability } \frac{1}{2} \\ -\frac{1}{\sqrt{d_\epsilon}} \text{ with probability } \frac{1}{2} \end{cases}$$

and

$$d_\epsilon = \left\lceil 92\epsilon^{-2} \max\left\{ 24K + 2K\log\left(\frac{\sqrt{K}}{\tau\epsilon^2}\right) + \log(2V_{\mathcal{M}}^2),\ \log\left(\frac{20}{\rho}\right) \right\} \right\rceil. \tag{6}$$

Then with probability at least $1 - \rho$ the following statement holds: for every pair of points $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{M}$,

$$(1 - \epsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2 \leq \|A\mathbf{x}_1 - A\mathbf{x}_2\|_2 \leq (1 + \epsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2.$$

Lemma 2 (Lemma 3 in [4]). *Let $f \in \mathcal{H}(\beta, E, M)$, where $0 < \beta \leq 1$, $M > 0$ and $E \subseteq [0,1]^d$ is a closed set with $d \in \mathbb{N}$. Then there exists a function $g \in \mathcal{H}(\beta, [0,1]^d, 2d^{\beta/2}M)$ such that $g(\mathbf{x}) = f(\mathbf{x})$ for any $\mathbf{x} \in E$.*

Our proof proceeds through the following 4 stages.

1. *Global dimension reduction.* A manifold JL lemma (Lemma 1) provides a nearly isometric linear map $A : \mathbb{R}^D \to \mathbb{R}^{d_e}$. After an affine rescaling, this yields $\Psi : \mathcal{M} \to [0,1]^{d_e}$ whose distortion is controlled by intrinsic geometry.
2. *Intrinsic Hölder representation and extension.* We pull back f to a reduced function on $\Psi(\mathcal{M})$ and extend it to $[0,1]^{d_e}$ via an extension lemma (Lemma 2).
3. *Grid approximation.* We approximate the extended function by a grid-based piecewise approximation $\widehat{h}(\mathbf{y}) = \sum_{\mathbf{m}} \rho_{\mathbf{m}}(\mathbf{y}) h(\mathbf{m}/K)$ where $\rho_{\mathbf{m}}$ is a tensor-product partition of unity built from a one-dimensional trapezoidal hat function ψ.
4. *Exact realization by a transformer.* We encode the grid indices $\mathbf{m}$ through positional embeddings, use a constant-size FFN to compute the hat function values $\psi(3K(y_i - m_i/K))$ tokenwise, and then use a sequence of attention blocks to (a) multiply these coordinatewise factors into the product weight $\rho_{\mathbf{m}}$ and (b) attach the corresponding sample value $h(\mathbf{m}/K)$. A final decoding layer sums the contributions.

Proof of Theorem 1. By Lemma 1, there exists a matrix $A \in \mathbb{R}^{d_\epsilon \times D}$, with d_ϵ given by (6), such that for every pair of points $\mathbf{x}_1$, $\mathbf{x}_2 \in \mathcal{M}$, we have

$$(1-\epsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2 \le \|A\mathbf{x}_1 - A\mathbf{x}_2\|_2 \le (1+\epsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2. \tag{7}$$

For any $\mathbf{x}_1$, $\mathbf{x}_2 \in \mathcal{M}$, using inequality (7), we have that

$$\begin{aligned} ||A\mathbf{x}_1 - A\mathbf{x}_2||_\infty &\overset{(i)}{\le} \|A\mathbf{x}_1 - A\mathbf{x}_2\|_2 \\ &\le (1+\epsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2 \\ &\le (1+\epsilon)\mathrm{diam}(\mathcal{M}) \\ &\overset{(ii)}{\le} 2\,\mathrm{diam}(\mathcal{M}), \end{aligned} \tag{8}$$

where (i) follows from the observation that, for any $\mathbf{z} = (z_1, \cdots, z_{d_\epsilon})^\top \in \mathbb{R}^{d_\epsilon}$, we have $|z_i| \le \sqrt{z_1^2 + \cdots + z_{d_\epsilon}^2}$ for $i = 1, \cdots, d_\epsilon$; inequality (ii) follows from the assumption that $\epsilon \le 1/3$.

Next, we use the matrix A to construct an affine transformation $\Psi : \mathcal{M} \mapsto [0,1]^{d_\epsilon}$ as

$$\Psi(\mathbf{x}) := \frac{1}{4\,\mathrm{diam}(\mathcal{M})} A\,\mathbf{x} - \frac{1}{4\,\mathrm{diam}(\mathcal{M})}\mathbf{y}_0, \tag{9}$$

where we choose $\mathbf{y}_0 \in \mathbb{R}^{d_\epsilon}$ such that $A(\mathcal{M}) \subseteq \{4\,\mathrm{diam}(\mathcal{M})\mathbf{y} + \mathbf{y}_0 \mid \mathbf{y} \in [0,1]^{d_\epsilon}\}$ and, for any $\mathbf{x} \in \mathcal{M}$. By construction, we have that $\Psi(\mathcal{M}) \subseteq [0,1]^{d_\epsilon}$.

By (7) and (9), we have

$$\frac{1-\epsilon}{4\,\mathrm{diam}(\mathcal{M})}\|\mathbf{x}_1 - \mathbf{x}_2\|_2 \le \|\Psi(\mathbf{x}_1) - \Psi(\mathbf{x}_2)\|_2 \le \frac{1+\epsilon}{4\,\mathrm{diam}(\mathcal{M})}\|\mathbf{x}_1 - \mathbf{x}_2\|_2. \tag{10}$$

We will next define a unique low-dimensional function g, with values on $[0,1]^{d_\epsilon}$, to represent the function f defined on $\mathcal{M}$. For any $\mathbf{y} \in \Psi(\mathcal{M}) \subseteq [0,1]^{d_\epsilon}$, we define

$$g(\mathbf{y}) := f(\mathbf{x}_\mathbf{y}), \text{ where } \mathbf{x}_\mathbf{y} = \{\mathbf{x} \in \mathcal{M} \,|\, \Psi(\mathbf{x}) = \mathbf{y}, \mathbf{y} \in \Psi(\mathcal{M})\}. \tag{11}$$

Note that, by inequality (10), the map Ψ is injective and, hence, it is bijective from $\mathcal{M}$ onto $\Psi(\mathcal{M})$.

We claim that the function g defined by (11) is a Hölder continuous function.

To show that this is the case, we first observe that, by the hypothesis of Theorem 1, the norm of f in (11) is bounded by M. It follows that

$$|g(\mathbf{y})| \leq M, \text{ for any } \mathbf{y} \in \Psi(\mathcal{M}). \tag{12}$$

Next, we observe that, for $\mathbf{y}_1, \mathbf{y}_2 \in \Psi(\mathcal{M})$, there are $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{M}$ defined by $\mathbf{x}_i = \{\mathbf{x} \in \mathcal{M} \mid \Psi(\mathbf{x}_i) = \mathbf{y}_i\}$, $i = 1, 2$. Using (10), it follows that

$$\begin{aligned} & |g(\mathbf{y}_1) - g(\mathbf{y}_2)| \\ = \; & |f(\mathbf{x}_1) - f(\mathbf{x}_2)| \\ \leq \; & M\|\mathbf{x}_1 - \mathbf{x}_2\|_2^\beta \\ \leq \; & M\left(\left(\tfrac{4\,\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta \|\Psi(\mathbf{x}_1) - \Psi(\mathbf{x}_2)\|_2^\beta\right) \\ \leq \; & \left(\tfrac{4\,\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta M\|\mathbf{y}_1 - \mathbf{y}_2\|_2^\beta. \end{aligned} \tag{13}$$

Combining (12) with (13), we conclude that $g \in \mathcal{H}\left(\beta, \Psi(\mathcal{M}), \left(\tfrac{4\,\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta M\right)$.

Using Lemma 2, we now extend the function g, originally defined on $\Psi(\mathcal{M})$, to a Hölder function h defined on $[0,1]^{d_\epsilon}$. Note that the Hölder norm of h is bounded by $8\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta {d_\epsilon}^{\beta/2} M$ and, thus, h belongs to the Hölder space

$$\mathcal{H}\left(\beta, [0,1]^{d_\epsilon}, 8\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta {d_\epsilon}^{\beta/2} M\right).$$

We next show that we can approximate h. First, we set

$$h_+ = h + 8\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta {d_\epsilon}^{\beta/2} M,$$

which implies

$$h_+ > 0 \text{ and } h_+ \in \mathcal{H}\left(\beta, [0,1]^{d_\epsilon}, 16\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^\beta {d_\epsilon}^{\beta/2} M\right).$$

Let $K \in \mathbb{N}$. Consider a partition of unity formed by a grid of $(K+1)^{d_\epsilon}$ functions $\rho_{\mathbf{m}}$ on $[0,1]^{d_\epsilon}$:

$$\sum_{\mathbf{m}} \rho_{\mathbf{m}}(\mathbf{y}) \equiv 1, \quad \mathbf{y} \in [0,1]^{d_\epsilon},$$

where $\mathbf{m} = (m_1, \cdots, m_{d_\epsilon}) \in \{0, 1, \cdots, K\}^{d_\epsilon}$, and the function $\rho_{\mathbf{m}}$ is defined as follows:

$$\rho_{\mathbf{m}}(\mathbf{y}) = \prod_{i=1}^{d_\epsilon} \psi\left(3K\left(y_i - \frac{m_i}{K}\right)\right), \tag{14}$$

where

$$\psi(t) = \begin{cases} 1, & |t| < 1 \\ 2 - |t|, & 1 \leq |t| \leq 2 \\ 0, & |t| > 2 \end{cases}. \tag{15}$$

We observe that

$$\|\rho_{\mathbf{m}}\|_\infty = 1 \tag{16}$$

and the support of $\rho_{\mathbf{m}}$ is contained in the set

$$\left\{\mathbf{y} : |y_i - \tfrac{m_i}{K}| \le \tfrac{2}{3K},\, i = 1, \cdots, d_\epsilon\right\} \subset \left\{\mathbf{y} : |y_i - \tfrac{m_i}{K}| \le \tfrac{1}{K},\, i = 1, \cdots, d_\epsilon\right\}.$$

Using the partition of unity (14), we decompose the target function

$$h_+ \in \mathcal{H}\left(\beta, [0,1]^{d_\epsilon}, 16\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^{\beta} {d_\epsilon}^{\beta/2} M\right)$$

as:

$$h_+(\mathbf{y}) = \sum_{\mathbf{m}} \rho_{\mathbf{m}}(\mathbf{y})\, h_+(\mathbf{y}). \tag{17}$$

The approximation for h_+ is as follows:

$$\widehat{h}_+(\mathbf{y}) = \sum_{\mathbf{m}} \rho_{\mathbf{m}}(\mathbf{y})\, h_+(\tfrac{\mathbf{m}}{K}). \tag{18}$$

Also, we will write the approximation for g on $\Psi(\mathcal{M}) \subset [0,1]^{d_\epsilon}$ as follows:

$$\widehat{g}(\mathbf{y}) = \sum_{\mathbf{m}} \rho_{\mathbf{m}}(\mathbf{y})\, g(\tfrac{\mathbf{m}}{K}). \tag{19}$$

We note that $\widehat{h}_+ = \widehat{g} + c^*$, where

$$c^* := 8\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^{\beta} {d_\epsilon}^{\beta/2} M. \tag{20}$$

By (17) and (18), the total approximation error is bounded by

$$\begin{aligned}
\max_{\mathbf{y}\in\Psi(\mathcal{M})} |\widehat{g}(\mathbf{y}) - g(\mathbf{y})| &= \max_{\mathbf{y}\in\Psi(\mathcal{M})} |(\widehat{g}(\mathbf{y}) + c^*) - (g(\mathbf{y}) + c^*)| \\
&\le \max_{\mathbf{y}\in[0,1]^{d_\epsilon}} \left|\widehat{h}_+(\mathbf{y}) - h_+(\mathbf{y})\right| \\
&= \max_{\mathbf{y}\in[0,1]^{d_\epsilon}} \left|\widehat{h}_+(\mathbf{y}) - h_+(\mathbf{y})\right| \\
&= \max_{\mathbf{y}\in[0,1]^{d_\epsilon}} \sum_{\mathbf{m}} \left|\rho_{\mathbf{m}}(\mathbf{y}) f(\tfrac{\mathbf{m}}{K}) - \rho_{\mathbf{m}}(\mathbf{y}) f(\mathbf{y})\right| \\
&\le \|\rho_{\mathbf{m}}\|_\infty \max_{\mathbf{y}\in[0,1]^{d_\epsilon}} \sum_{\mathbf{m}} \left|f(\tfrac{\mathbf{m}}{K}) - f(\mathbf{y})\right| \\
&\overset{(i)}{=} \max_{\mathbf{y}\in[0,1]^{d_\epsilon}} \sum_{\left\{\mathbf{m}:\|\mathbf{y}-\frac{\mathbf{m}}{K}\|_\infty \le \frac{1}{K}\right\}} \left|f(\tfrac{\mathbf{m}}{K}) - f(\mathbf{y})\right| \\
&\overset{(ii)}{\le} 16\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^{\beta} {d_\epsilon}^{\beta/2} M \max_{\mathbf{y}\in[0,1]^{d_\epsilon}} \sum_{\left\{\mathbf{m}:\|\mathbf{y}-\frac{\mathbf{m}}{K}\|_\infty \le \frac{1}{K}\right\}} \left\|\tfrac{\mathbf{m}}{K} - \mathbf{y}\right\|_2 \\
&\overset{(iii)}{\le} 2^{d_\epsilon} \times (\tfrac{\sqrt{d_\epsilon}}{K})^{\beta} \times 16\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^{\beta} {d_\epsilon}^{\beta/2} M,
\end{aligned} \tag{21}$$

where (i) uses $\|\rho_{\mathbf{m}}\|_\infty = 1$ in (16); (ii) follows from

$$h_+ \in \mathcal{H}\left(\beta, [0,1]^{d_\epsilon}, 16\left(\frac{\mathrm{diam}(\mathcal{M})}{1-\epsilon}\right)^{\beta} {d_\epsilon}^{\beta/2} M\right);$$

(iii) is obtained by the observation $\|\mathbf{y} - \frac{m}{K}\|_2 \le \frac{\sqrt{d}}{K}$.

From (21), we can control the total error by increasing K. In the following, our task is to construct an appropriate transformer neural network to realize Ψ defined in (9) and $\widehat{g}$ given by (19) exactly.

Step 1: Embed the input

Using the matrix $A \in \mathbb{R}^{d_\epsilon \times D}$ defined in above (7), we construct the following embedding layer $E(\mathbf{x}) \in \mathbb{R}^{5\times l}$ defined in (3):

$$E(\mathbf{x}) \;=\; r\left(\frac{1}{4\,\mathrm{diam}(\mathcal{M})}A\mathbf{x}\right)^{\top} S, \tag{22}$$

where $l = d_\epsilon K^{d_\epsilon}$ is the number of hidden tokens; $r = (1,0,0,0,0)^\top \in \mathbb{R}^5$;

$$S = \left[\begin{array}{ccccc|c|ccccc} 1 & 0 & 0 & \cdots & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{array}\right] \in \mathbb{R}^{d_\epsilon \times l}.$$

We next build the positional encoding matrix:

$$P = \left[\begin{array}{c|c|c|c|c} \frac{-\mathbf{y}_0^\top}{4\,\mathrm{diam}(\mathcal{M})} & \frac{-\mathbf{y}_0^\top}{4\,\mathrm{diam}(\mathcal{M})} & \cdots & \frac{-\mathbf{y}_0^\top}{4\,\mathrm{diam}(\mathcal{M})} & \frac{-\mathbf{y}_0^\top}{4\,\mathrm{diam}(\mathcal{M})} \\ \frac{\mathbf{m}_1^\top}{K} & \frac{\mathbf{m}_2^\top}{K} & \cdots & \frac{\mathbf{m}_{K^{d_\epsilon}-1}^\top}{K} & \frac{\mathbf{m}_{K^{d_\epsilon}}^\top}{K} \\ \mathbf{1}_{d_\epsilon}^\top & \mathbf{1}_{d_\epsilon}^\top & \cdots & \mathbf{1}_{d_\epsilon}^\top & \mathbf{1}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top & \cdots & \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top & \cdots & \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top \end{array}\right] \in \mathbb{R}^{5\times l},$$

where $\mathbf{y}_0 \in \mathbb{R}^{d_\epsilon}$ is defined in (9); $\mathbf{m}_1, \ldots, \mathbf{m}_{K^{d_\epsilon}} \in \{0,\ldots,K\}^{d_\epsilon}$ are different vectors; $\mathbf{1}_{d_\epsilon} := (1,1,\ldots,1)^\top \in \mathbb{R}^{d_\epsilon}$; $\mathbf{0}_{d_\epsilon} := (0,0,\ldots,0)^\top \in \mathbb{R}^{d_\epsilon}$.

Now, using E in (22), we get the following input embedding matrix:

$$H_1 = E(\mathbf{x}) + P = \left[\begin{array}{c|c|c|c|c} \mathbf{y}^\top & \mathbf{y}^\top & \cdots & \mathbf{y}^\top & \mathbf{y}^\top \\ \frac{\mathbf{m}_1^\top}{K} & \frac{\mathbf{m}_2^\top}{K} & \cdots & \frac{\mathbf{m}_{K^{d_\epsilon}-1}^\top}{K} & \frac{\mathbf{m}_{K^{d_\epsilon}}^\top}{K} \\ \mathbf{1}_{d_\epsilon}^\top & \mathbf{1}_{d_\epsilon}^\top & \cdots & \mathbf{1}_{d_\epsilon}^\top & \mathbf{1}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top & \cdots & \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top & \cdots & \mathbf{0}_{d_\epsilon}^\top & \mathbf{0}_{d_\epsilon}^\top \end{array}\right] \in \mathbb{R}^{5\times l}, \tag{23}$$

where $\mathbf{y} = \Psi(\mathbf{x}) \in [0,1]^{d_\epsilon}$ is defined in (9).

Step 2: Compute $\psi\left(3K\left(y_j - \frac{m_{ij}}{K}\right)\right)$ from (14)

We use the first FFN to compute the trapezoidal basis function $\psi(t) = \max(0, \min(1, 2-|t|)) = \varrho(t+2) - \varrho(t+1) - \varrho(t-1) + \varrho(t-2)$ given by (15) where $t = 3K(y_j - \frac{m_{ij}}{K})$.

Block 1. We skip Attention in this block (set $V = 0$), so $H_{1.5} = H_1$. We construct $W_{11} \in \mathbb{R}^{5\times 5}$ to generate the four terms:

$$W_{11} = 3K\begin{bmatrix} 1 & -1 & 2/(3K) & 0 & 0 \\ 1 & -1 & 1/(3K) & 0 & 0 \\ 1 & -1 & -1/(3K) & 0 & 0 \\ 1 & -1 & -2/(3K) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad b_{11} = \mathbf{0}_{5\times l} \tag{24}$$

This generates pre-activations $[t+2, t+1, t-1, t-2, 0]^\top$. After ϱ, we use $W_2 \in \mathbb{R}^{5\times 5}$ to combine them into Row 4 (Value):

$$W_{12} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } b_{12} = \mathbf{0}_{5\times l}.$$

$$W_{13} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and

$$b_{13} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right],$$

where

$$b_{13}[B_1] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ \frac{-m_{11}}{K} & \frac{-m_{12}}{K}+1 & \frac{-m_{13}}{K} & \dots & \frac{-m_{1d_\epsilon}}{K} \\ -1 & -1 & -1 & \dots & -1 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & -c & -c & \dots & -c \end{bmatrix} \in \mathbb{R}^{5\times d_\epsilon}$$

and

$$b_{13}[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \frac{-\mathbf{m}_i^\top}{K} \\ -\mathbf{1}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ -c\cdot\mathbf{1}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5\times d_\epsilon} \text{ for } i = 2, \dots, K^{d_\epsilon}$$

with

$$c = \max\left\{2,\ 16\left(\frac{\operatorname{diam}(\mathcal{M})}{1-\epsilon}\right)^{\beta} {d_\epsilon}^{\beta/2} M\right\}. \tag{25}$$

After using feed-forward network $FFN_1 \in \mathcal{FFN}(5,3)$ we designed above, we can get

$$FFN_1(H_{1.5}) = W_{13}\varrho(W_{12}\varrho(W_{11}H_{1.5} + b_{11}) + b_{12}) + b_{13}$$

$$= \left[\begin{array}{c|c|c|c|c} B_1 & \begin{matrix} -\mathbf{y}^\top \\ \frac{-\mathbf{m}_2^\top}{K} \\ -\mathbf{1}_{d_\epsilon}^\top \\ G_2 \\ \mathbf{0}_{d_\epsilon}^\top \end{matrix} & \begin{matrix} \dots \\ \dots \\ \dots \\ \dots \\ \dots \end{matrix} & \begin{matrix} -\mathbf{y}^\top \\ \frac{-\mathbf{m}_{K^{d_\epsilon}-1}^\top}{K} \\ -\mathbf{1}_{d_\epsilon}^\top \\ G_{K^{d_\epsilon}-1} \\ \mathbf{0}_{d_\epsilon}^\top \end{matrix} & \begin{matrix} -\mathbf{y}^\top \\ \frac{-\mathbf{m}_{K^{d_\epsilon}}^\top}{K} \\ -\mathbf{1}_{d_\epsilon}^\top \\ G_{K^{d_\epsilon}} \\ \mathbf{0}_{d_\epsilon}^\top \end{matrix} \end{array} \right]$$

with

$$FFN_1(H_{1.5})[B_1] = \begin{bmatrix} y_1 & y_2 & y_3 & \dots & y_{d_\epsilon} \\ \frac{-m_{11}}{K} & \frac{-m_{12}}{K}+1 & \frac{-m_{13}}{K} & \dots & \frac{-m_{1d_\epsilon}}{K} \\ -1 & -1 & -1 & \dots & -1 \\ (G_1)_1 & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} \in \mathbb{R}^{5\times d_\epsilon},$$

where for each $i = 1, \dots, K^{d_\epsilon}$ associated with $\mathbf{m}_i = (m_{i1}, \dots, m_{id_\epsilon})^\top \in \{0, \dots, K\}^{d_\epsilon}$, the row vector $G_i \in \mathbb{R}^{d_\epsilon}$ is defined by

$$G_i = \left(\psi(3K(y_1 - \tfrac{m_{i1}}{K})), \dots, \psi(3K(y_{d_\epsilon} - \tfrac{m_{id_\epsilon}}{K}))\right),$$

equivalently, its j-th entry satisfies

$$(G_i)_j = \psi\left(3K\left(y_j - \frac{m_{ij}}{K}\right)\right), \qquad j = 1, \dots, d_\epsilon,$$

By the definition of ψ in (15), we have

$$0 \le (G_i)_j \le 1, \tag{26}$$

which also implies

$$\prod_{j=1}^{k_1}(G_i)_j \ge \prod_{j=1}^{k_2}(G_i)_j \ge 0 \text{ if } k_1 \le k_2. \tag{27}$$

According to (25) and (26), we have

$$\varrho\left(\prod_{j=1}^{k}(G_i)_j - c\right) = 0 \text{ for } k = 1, \dots, d_\epsilon, \tag{28}$$

where ϱ is the ReLU activation function. Note that we will repeatedly use this observation in the following.

This produces the next embedding matrix H_2:

$$H_2 = FFN_1(H_{1.5}) + H_{1.5} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array}\right],$$

where

$$H_2[B_1] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ (G_1)_1 & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 0 & -c & -c & \dots & -c \end{bmatrix} \in \mathbb{R}^{5 \times d_\epsilon}$$

and

$$H_2[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ G_i \\ -c \cdot \mathbf{1}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5 \times d_\epsilon} \text{ for } i = 2, \dots, K^{d_\epsilon}$$

with c defined in (25).

Step 3: Compute $\rho_{\mathbf{m}}$ defined in (14)

Block 2. Define Q_1, K_1 and V_1 of the second transformer block as follows:

$$Q_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad K_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } V_1 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

We can get the following $H_{2.5}$:

$$H_{2.5} = V_1 H_2 \varrho((K_1 H_2)^\top Q_1 H_2) + H_2 = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right], \tag{29}$$

where

$$H_{2.5}[B_1] = \begin{bmatrix} (G_1)_1 \cdot (G_1)_2 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ (G_1)_1 & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 0 & -c & -c & \dots & -c \end{bmatrix},$$

and

$$H_{2.5}[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ G_i \\ -c \cdot \mathbf{1}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5 \times d_\epsilon} \text{ for } i = 2, \dots, K^{d_\epsilon}.$$

We next construct a feed-forward network $FFN_2 \in \mathcal{FFN}(5,2)$ defined as:

$$FFN_2(H_{2.5}) = W_{22}\varrho(W_{21}H_{2.5} + b_{21}) + b_{22}$$

with

$$W_{21} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } b_{21} = \mathbf{0}_{5 \times l} \tag{30}$$

and

$$W_{22} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } b_{22} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right] \tag{31}$$

with

$$b_{22}[B_1] = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & -1 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

and

$$b_{22}[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5 \times d_\epsilon} \text{ for } i = 2, \dots, K^{d_\epsilon}.$$

By (27) and (28), we obtain that

$$FFN_2(H_{2.5}) = W_{22}\varrho(W_{21}H_{2.5} + b_{21}) + b_{22} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right],$$

where

$$FFN_2(H_{2.5})[B_1] = \begin{bmatrix} -(G_1)_1 \cdot (G_1)_2 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ (G_1)_1 \cdot (G_1)_2 - (G_1)_1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix},$$

and

$$FFN_2(H_{2.5})[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5\times d_\epsilon} \text{ for } i = 2, \dots, K^{d_\epsilon}.$$

Then we can get the next embedding matrix:

$$H_3 = FNN_2(H_{2.5}) + H_{2.5} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right],$$

where

$$H_3[B_1] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ (G_1)_1 \cdot (G_1)_2 & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 0 & -c & -c & \dots & -c \end{bmatrix}$$

and

$$H_3[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ G_i \\ -c \cdot \mathbf{1}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5\times d_\epsilon} \text{ for } i = 2, \dots, K^{d_\epsilon}.$$

Block 3 – Block $d_\epsilon K^{d_\epsilon} + 1$. From Block 3 to Block $d_\epsilon K^{d_\epsilon} + 1$, the Q, K and V of each block are identical to those of Block 2, as specified in (29). The only difference lies in the second layer of the FFN layer, which is constructed as follows.

For $s = 0, \dots, K^{d_\epsilon}$, the FFNs of Blocks $sd_\epsilon + 2$ to $sd_\epsilon + d_\epsilon$ differ from the FFN of Block 2 only in the bias term b_{22}, i.e., for $s = 0, \dots, K^{d_\epsilon} - 1$ and $i = 2, \dots, d_\epsilon$,

$$W_{(sd_\epsilon+i)\,1} = W_{21}, \quad b_{(sd_\epsilon+i)\,1} = b_{21}, \text{ and } W_{(sd_\epsilon+i)\,2} = W_{22},$$

where W_{21} and b_{21} are defined in (30); W_{22} and b_{22} are given by (31).

More precisely, for Block $sd_\epsilon + i$ with $s = 0, \dots, K^{d_\epsilon} - 1$ and $i = 2, \dots, d_\epsilon$, the bias vector $b_{(sd_\epsilon+i)\,2}$ satisfies

$$(b_{(sd_\epsilon+i)\,2})_{2,i} = -1, \quad (b_{(sd_\epsilon+i)\,2})_{2,i+1} = 1,$$

and all other entries are zero.

Moreover, for Block $sd_\epsilon + 1$ with $s = 1, \dots, K^{d_\epsilon} - 1$, the bias vector $b_{(sd_\epsilon+1)\,2}$ has the following nonzero entries:

$$(b_{(sd_\epsilon+1)\,2})_{2,sd_\epsilon} = -1,\ (b_{(sd_\epsilon+1)\,2})_{2,sd_\epsilon+2} = 1,\ (b_{(sd_\epsilon+1)\,2})_{5,sd_\epsilon-d+1} = -c,\ (b_{(sd_\epsilon+1)\,2})_{5,sd_\epsilon+1} = c,$$

with all remaining entries equal to zero.

We construct the FFN layer of Block $l+1$ with $l = d_\epsilon K^{d_\epsilon}$ with bias matrix $b_{(ld_\epsilon+1)\,2}$ as follows:

$$(b_{(ld_\epsilon+1)\,2})_{2,ld_\epsilon} = -1,$$

and for the fifth row,

$(b_{(ld_\epsilon+1)\,2})_{5,1} = c+1$, $(b_{(ld_\epsilon+1)\,2})_{5,ld_\epsilon-d_\epsilon+1} = 0$, and $(b_{(ld_\epsilon+1)\,2})_{5,j} = c$ for all other columns j.

All remaining entries of $b_{(ld_\epsilon+1)\,2}$ are set to zero.

By applying the above procedure from Block 1 to Block $l+1$, we obtain the following embedding matrix:

$$H_{l+2} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array}\right],$$

where

$$H_{l+2}[B_1] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_1} & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

and

$$H_{l+2}[B_i] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_i} & (G_i)_2 & (G_i)_3 & \dots & (G_i)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

for $i = 2, \dots, K^{d_\epsilon}$.

Step 4: Compute $\rho_{\mathbf{m}}(\mathbf{y}) h_+(\frac{\mathbf{m}}{K})$ defined in (18)

Block $l+2$**.** Define Q_2, K_2 and V_2 of the second transformer block as follows:

$$Q_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \; K_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_+(\frac{\mathbf{m}_1}{K}) & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } V_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

We can get the following $H_{2.5}$:

$$H_{l+2.5} = V_2 H_{l+2} \varrho((K_2 H_{l+2})^\top Q_2 H_{l+2}) + H_{l+2} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array}\right], \quad (32)$$

where

$$H_{l+2.5}[B_1] = \begin{bmatrix} \rho_{\mathbf{m}_1} h_+(\frac{\mathbf{m}_1}{K}) & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_1} & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix},$$

and

$$H_{l+2.5}[B_i] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_i} & (G_i)_2 & (G_i)_3 & \dots & (G_i)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

for $i = 2, \dots, K^{d_\epsilon}$.

We next construct a feed-forward network $FFN_{l+2} \in \mathcal{FFN}(5,1)$ defined as:

$$FFN_{l+2}(H_{l+2.5}) = W_{(l+2)\,1} H_{l+2.5} + b_{(l+2)\,1}$$

with

$$W_{(l+2)\,1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } b_{(l+2)\,1} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right]$$

with

$$b_{(l+2)\,1}[B_1] = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ -1 & 0 & 0 & 0 & \dots & 0 \end{bmatrix}, \quad b_{(l+2)\,1}[B_2] = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

and

$$b_{(l+2)\,1}[B_i] = \begin{bmatrix} \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \\ \mathbf{0}_{d_\epsilon}^\top \end{bmatrix} \in \mathbb{R}^{5 \times d_\epsilon} \text{ for } i = 3, \dots, K^{d_\epsilon}.$$

Then we can get the next embedding matrix:

$$H_{l+3} = FNN_{l+2}(H_{l+2.5}) + H_{l+2.5} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array} \right],$$

where

$$H_{l+3}[B_1] = \begin{bmatrix} \rho_{\mathbf{m}_1} h_+(\frac{\mathbf{m}_1}{K}) & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_1} & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix},$$

$$H_{l+3}[B_2] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_2} & (G_2)_2 & (G_2)_3 & \dots & (G_2)_{d_\epsilon} \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

and

$$H_{l+3}[B_i] = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_i} & (G_i)_2 & (G_i)_3 & \dots & (G_i)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} \text{ for } i = 3, \dots, K^{d_\epsilon}.$$

Block $l+3$ – Block $l+K^{d_\epsilon}+1$. From Block $l+3$ to Block $l+K^{d_\epsilon}+1$, the Q and V of each block are identical to those of Block $l+2$, as specified in (32). The only difference lies in K and the FFN layer, which is constructed as follows.

In Block $l+i$ with $i = 3, \dots, K^{d_\epsilon}+1$,

$$K = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_+(\frac{\mathbf{m}_{i-1}}{K}) & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

For $i = 3, \dots, K^{d_\epsilon}$, the FFNs of Blocks $l+3$ to $l+K^{d_\epsilon}$ differ from the FFN of Block $l+2$ only in the bias term $b_{(l+2)\,1}$. More precisely, for Block $l+i$ with $i = 3, \dots, K^{d_\epsilon}+1$, the bias vector $b_{(l+i)\,1}$ satisfies

$$(b_{(l+i)\,1})_{5,\,(i-2)d_\epsilon+1} = -1, \qquad (b_{(l+i)\,1})_{5,\,(i-1)d_\epsilon+1} = 1,$$

and all other entries are zero.

Moreover, for Block $l+K^{d_\epsilon}+1$, the bias vector $b_{(l+K^{d_\epsilon}+1)\,1}$ has the following nonzero entries:

$$(b_{(l+K^{d_\epsilon}+1)\,1})_{1,1} = -c^* \text{ and } (b_{(l+K^{d_\epsilon}+1)\,1})_{5,l-d_\epsilon+1} = -1$$

with all remaining entries equal to zero, where c^* is defined in (20).

By applying the above procedure from Block $l+2$ to Block $l+K^{d_\epsilon}+1$, we obtain the following embedding matrix:

$$H_{l+K^{d_\epsilon}+1} = \left[\begin{array}{c|c|c|c|c} B_1 & B_2 & \cdots & B_{K^{d_\epsilon}-1} & B_{K^{d_\epsilon}} \end{array}\right],$$

where

$$H_{l+K^{d_\epsilon}+1}[B_1] = \begin{bmatrix} \rho_{\mathbf{m}_1} h_+(\frac{\mathbf{m}_i}{K}) - c^* & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_1} & (G_1)_2 & (G_1)_3 & \dots & (G_1)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

and

$$H_{l+K^{d_\epsilon}+1}[B_i] = \begin{bmatrix} \rho_{\mathbf{m}_i} h_+(\frac{\mathbf{m}_i}{K}) & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \rho_{\mathbf{m}_i} & (G_i)_2 & (G_i)_3 & \dots & (G_i)_{d_\epsilon} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

for $i = 2, \dots, K^{d_\epsilon}$, where c^* is given by (20).

Using the decoding head $\mathcal{D}$ defined (2) on $H_{l+K^{d_\epsilon}+1}$, we can get the exact expression of $\widehat{g}$ given by (19).

We finally show that, for an appropriate choice of ρ, we can express d_ϵ in terms of ϵ, d, $V_{\mathcal{M}}$ and τ. In fact, by taking

$$\rho = \frac{10\tau^{2d}\epsilon^{4d}}{(e^{24}d)^d V_{\mathcal{M}}^2},$$

a direct calculation gives the following equality

$$\left(24d + 2d\log\left(\frac{\sqrt{d}}{\tau\epsilon^2}\right) + \log(2V_{\mathcal{M}}^2)\right) = \log\left(\frac{20}{\rho}\right),$$

so that, by (6), we have

$$d_\epsilon \quad = \quad \left\lceil 92\epsilon^{-2}\left(24d + 2d\log\left(\frac{\sqrt{d}}{\tau\epsilon^2}\right) + \log(2V_{\mathcal{M}}^2)\right)\right\rceil,$$

where there is no dependence on ρ. Additionally, by the condition $\frac{V_{\mathcal{M}}}{\tau^d} \geq \left(\frac{21}{2\sqrt{d}}\right)^d$ in the hypothesis, it follows that

$$\rho = \frac{10\tau^{2d}\epsilon^{4d}}{(e^{24}d)^d V_{\mathcal{M}}^2} \leq 10\left(\frac{4\epsilon^4}{441e^{24}}\right)^d,$$

showing that ρ is very small. The proof is completed by choosing $\epsilon = 1/3$ and identifying $d_e = d_{1/3}$, as in (4). Finally, setting $N = K^{d_e}$ in (21) yields an error bound, $3\left(2^{d_e+3}d^\beta M \operatorname{diam}(\mathcal{M})\right)N^{-\frac{\beta}{d_e}}$. □

4. Numerical Experiment

In this section, we present a numerical experiment to complement Theorem 1 stating that the approximation estimate takes the form

$$\max_{\mathbf{x}\in\mathcal{M}} |T(\mathbf{x}) - f(\mathbf{x})| \leq C\, N^{-\beta/d_e},$$

where the constant C is independent of the ambient dimension D. The purpose of the following experiment is to illustrate precisely this ambient-dimension-independent behavior. To isolate the role of D, we consider the same circle embedded in different ambient dimensions

$$\mathcal{M}_D = \{(0.5 + 0.35\cos\theta,\ 0.5 + 0.35\sin\theta,\ 0, \ldots, 0) : \theta \in [0, 2\pi)\} \subset [0,1]^D,$$

for $D = 2, 4, 8, 16, 32, 64, 128, 256, 512$, and the scalar target

$$f(\theta) = \sin(3\theta) + 0.3\cos(5\theta).$$

For this family of zero-padded embeddings of the same circle, the intrinsic geometry is unchanged as D varies, and hence the effective dimension d_e defined in (4) is also independent of D. For every D, we use the same single-head ReLU-transformer architecture of Definition 2 with a trainable embedding matrix U. In particular, the resolution parameter is fixed at $N = 2$ for all ambient dimensions, so the factor $N^{-\beta/d_e}$ in the bound is the same throughout the experiment and the architecture size is unchanged. We also fix the embedding dimension at $d_{\text{embd}} = 5$, the tokenwise feed-forward depth at $L_{\text{FFN}} = 3$, the width at $W_{\text{FFN}} = 5$, the number of hidden tokens at four, and the number of transformer blocks at four. The optimization settings are also kept fixed across all dimensions. For each trained network T, we numerically approximate $\|T - f\|_{L^\infty(\mathcal{M}_D)}$ by evaluating $|T(x(\theta_j)) - f(\theta_j)|$ on the uniform grid $\theta_j = 2\pi j/2048$, $j = 0, \ldots, 2047$, and taking the maximum.

Figure 2 reports the mean numerically approximated L^∞ error over 20 random seeds. The curve remains nearly flat: the mean error stays between 0.2010 and 0.2223 as D ranges from 2 to 512. This supports the main result of Theorem 1: the approximation behavior is essentially unaffected by the ambient dimension and is governed instead by the low-dimensional structure of the data.

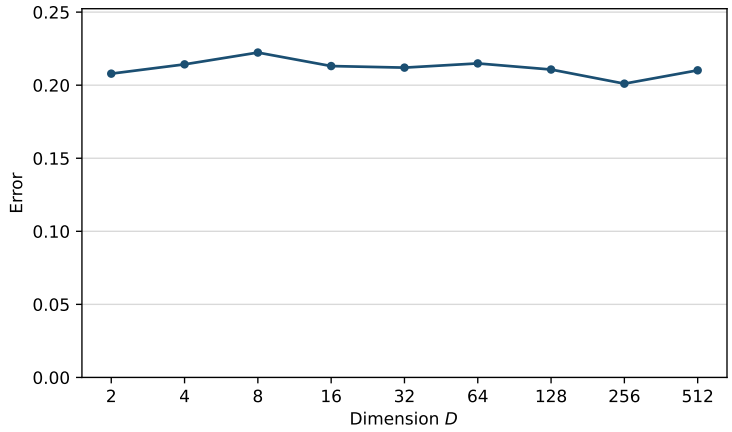

Figure 2. Mean numerically approximated L^∞ error versus the ambient dimension D for the same circle embedded in $\mathbb{R}^D$.

5. Conclusions

We proved a uniform approximation bound for transformers on $\mathcal{M} \subset [0,1]^D$ showing that not only the approximation rate but also the multiplicative constant is independent of the ambient dimension D. This result contributes to an emerging thrusts from the literature [4] aimed at providing a solid theoretical underpinning to the observation that deep learning architecture are remarkably effective to approximate high-dimensional data while seemingly bypassing the curse of dimensionality. This is the first result showing that transformer can achieves a fully dimension-independent approximation estimate in the manifold setting. To prove this result, we introduced new theoretical tools that are expected to be useful to advance conceptual understanding of the expressive power of deep architectures. The numerical experiment in Section 4 provides additional empirical support for this conclusion by showing that, for the same circle embedded in dimensions from 2 to 512, the observed error is nearly flat as the ambient dimension varies.

A limitation of the present work is that the analysis is restricted to a single-head ReLU-attention surrogate and addresses only approximation. Natural directions for future work include studying whether generalization error bounds with constants independent of the ambient dimension D can also be established, and whether analogous results hold for standard softmax attention.

Author Contributions: Conceptualization, D.L. and J.S.; methodology, D.L. and J.S.; formal analysis, D.L. and J.S.; writing—original draft preparation, D.L. and J.S.; writing—review and editing, D.L. and J.S.; funding acquisition, D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by Simons Foundation grant MPS-TSM-00002738.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Advances in neural information processing systems* **2017**, *30*.
2. Peebles, W.; Xie, S. Scalable diffusion models with transformers. In Proceedings of the Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 4195–4205.
3. Gorban, A.N.; Tyukin, I.Y. Blessing of dimensionality: mathematical foundations of the statistical physics of data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **2018**, *376*, 20170237.
4. Labate, D.; Shi, J. Low dimensional approximation and generalization of multivariate functions on smooth manifolds using deep ReLU neural networks. *Neural Networks* **2024**, *174*, 106223.
5. Shaham, U.; Cloninger, A.; Coifman, R.R. Provable approximation properties for deep neural networks. *Applied and Computational Harmonic Analysis* **2018**, *44*, 537–557.
6. Schmidt-Hieber, J. Deep ReLU network approximation of functions on a manifold. *arXiv preprint arXiv:1908.00695* **2019**.
7. Nakada, R.; Imaizumi, M. Adaptive Approximation and Generalization of Deep Neural Network with Intrinsic Dimensionality. *J. Mach. Learn. Res.* **2020**, *21*, 1–38.
8. Cloninger, A.; Klock, T. A deep network construction that adapts to intrinsic dimensionality beyond the domain. *Neural Networks* **2021**, *141*, 404–419.
9. Chen, M.; Jiang, H.; Liao, W.; Zhao, T. Nonparametric regression on low-dimensional manifolds using deep ReLU networks: Function approximation and statistical recovery. *Information and Inference: A Journal of the IMA* **2022**, *11*, 1203–1253.
10. Jiao, Y.; Shen, G.; Lin, Y.; Huang, J. Deep nonparametric regression on approximate manifolds: Nonasymptotic error bounds with polynomial prefactors. *The Annals of Statistics* **2023**, *51*, 691–716.
11. Cai, J.F.; Li, D.; Sun, J.; Wang, K. Enhanced Expressive Power and Fast Training of Neural Networks by Random Projections. *CSIAM Transactions on Applied Mathematics* **2021**, *2*, 532–550. https://doi.org/10.4208/csiam-am.SO-2020-0004.
12. Shen, Z.; Yang, H.; Zhang, S. Deep Network Approximation Characterized by Number of Neurons. *Communications in Computational Physics* **2020**, *28*, 1768–1811. https://doi.org/https://doi.org/10.4208/cicp.OA-2020-0149.
13. Eftekhari, A.; Wakin, M.B. New analysis of manifold embeddings and signal recovery from compressive measurements. *Applied and Computational Harmonic Analysis* **2015**, *39*, 67–109.
14. Oono, K.; Suzuki, T. Approximation and non-parametric estimation of ResNet-type convolutional neural networks. In Proceedings of the International conference on machine learning. PMLR, 2019, pp. 4922–4931.
15. Liu, H.; Chen, M.; Zhao, T.; Liao, W. Besov function approximation and binary classification on low-dimensional manifolds using convolutional residual networks. In Proceedings of the International Conference on Machine Learning. PMLR, 2021, pp. 6770–6780.
16. Kovachki, N.; Lanthaler, S.; Mishra, S. On universal approximation and error bounds for Fourier neural operators. *Journal of Machine Learning Research* **2021**, *22*, 1–76.
17. Chen, G.; Liu, X.; Meng, Q.; Chen, L.; Liu, C.; Li, Y. Learning neural operators on riemannian manifolds. *National Science Open* **2024**, *3*, 20240001.
18. Yun, C.; Bhojanapalli, S.; Rawat, A.S.; Reddi, S.J.; Kumar, S. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077* **2019**.
19. Yun, C.; Chang, Y.W.; Bhojanapalli, S.; Rawat, A.S.; Reddi, S.; Kumar, S. O (n) connections are expressive enough: Universal approximability of sparse transformers. *Advances in Neural Information Processing Systems* **2020**, *33*, 13783–13794.
20. Pérez, J.; Barceló, P.; Marinkovic, J. Attention is turing-complete. *Journal of Machine Learning Research* **2021**, *22*, 1–35.
21. Takakura, S.; Suzuki, T. Approximation and estimation ability of transformers for sequence-to-sequence functions with infinite dimensional input. In Proceedings of the International Conference on Machine Learning. PMLR, 2023, pp. 33416–33447.
22. Takeshita, N.; Imaizumi, M. Approximation of permutation invariant polynomials by transformers: Efficient construction in column-size. *arXiv preprint arXiv:2502.11467* **2025**.
23. Jiao, Y.; Lai, Y.; Sun, D.; Wang, Y.; Yan, B. Approximation Bounds for Transformer Networks with Application to Regression. *arXiv preprint arXiv:2504.12175* **2025**.
24. Jiang, H.; Li, Q. Approximation rate of the transformer architecture for sequence modeling. *Advances in Neural Information Processing Systems* **2024**, *37*, 68926–68955.
25. Kim, J.; Suzuki, T. Transformers provably solve parity efficiently with chain of thought. *URL https://arxiv. org/abs/2410.08633* **2024**.
26. Kim, J.; Nakamaki, T.; Suzuki, T. Transformers are minimax optimal nonparametric in-context learners. *Advances in Neural Information Processing Systems* **2024**, *37*, 106667–106713.

27. Havrilla, A.; Liao, W. Understanding scaling laws with statistical and approximation theory for transformer neural networks on intrinsically low-dimensional data. *Advances in Neural Information Processing Systems* **2024**, *37*, 42162–42210.
28. Lin, L.; Bai, Y.; Mei, S. Transformers as decision makers: Provable in-context reinforcement learning via supervised pretraining. *arXiv preprint arXiv:2310.08566* **2023**.
29. Bai, Y.; Chen, F.; Wang, H.; Xiong, C.; Mei, S. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Advances in neural information processing systems* **2023**, *36*, 57125–57211.
30. Lai, Z.; Lim, L.H.; Liu, Y. Attention is a smoothed cubic spline. *arXiv preprint arXiv:2408.09624* **2024**.
31. Shen, K.; Guo, J.; Tan, X.; Tang, S.; Wang, R.; Bian, J. A study on relu and softmax in transformer. *arXiv preprint arXiv:2302.06461* **2023**.
32. Hu, J.Y.C.; Liu, H.; Chen, H.Y.; Wu, W.; Liu, H. Universal approximation with softmax attention. *arXiv preprint arXiv:2504.15956* **2025**.
33. Niyogi, P.; Smale, S.; Weinberger, S. Finding the homology of submanifolds with high confidence from random samples. *Discrete & Computational Geometry* **2008**, *39*, 419–441.