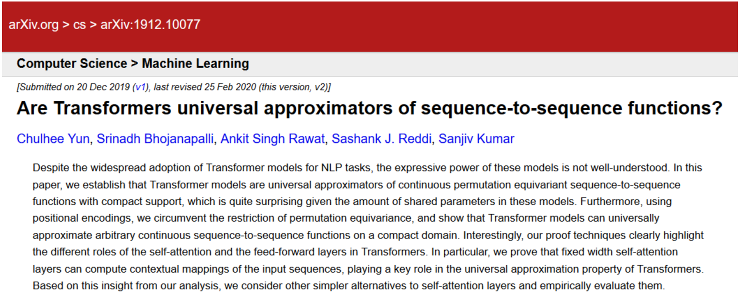
Transformers are universal approximators

Yun, Bhojanapalli, Rawat, Reddi, Kumar '20

Geelon So
(agso@eng.ucsd.edu)

April 22, 2020



arXiv.org > cs > arXiv:1912.10077

Computer Science > Machine Learning

[Submitted on 20 Dec 2019 (v1), last revised 25 Feb 2020 (this version, v2)]

Are Transformers universal approximators of sequence-to-sequence functions?

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, Sanjiv Kumar

Despite the widespread adoption of Transformer models for NLP tasks, the expressive power of these models is not well-understood. In this paper, we establish that Transformer models are universal approximators of continuous permutation equivariant sequence-to-sequence functions with compact support, which is quite surprising given the amount of shared parameters in these models. Furthermore, using positional encodings, we circumvent the restriction of permutation equivariance, and show that Transformer models can universally approximate arbitrary continuous sequence-to-sequence functions on a compact domain. Interestingly, our proof techniques clearly highlight the different roles of the self-attention and the feed-forward layers in Transformers. In particular, we prove that fixed width self-attention layers can compute contextual mappings of the input sequences, playing a key role in the universal approximation property of Transformers. Based on this insight from our analysis, we consider other simpler alternatives to self-attention layers and empirically evaluate them.

Introduction

Background: transformer networks

Transformer networks are a recent approach to learning *sequence-to-sequence* functions,

$$f : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}.$$

Background: interaction

The difficulty of learning seq-to-seq functions lies with the **interactions** between the *tokens* in the sequence.

Background: interaction

The difficulty of learning seq-to-seq functions lies with the **interactions** between the *tokens* in the sequence.

- If tokens don't interact in the computation of f, then we'd expect for some f_i's:

$$f(x_1, \ldots, x_n) = \big(f_1(x_1), \ldots, f_n(x_n)\big).$$

Background: interaction

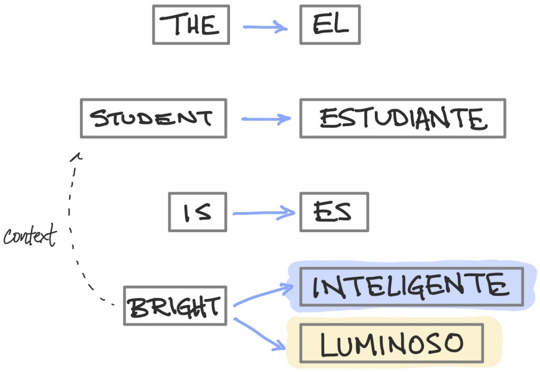


Figure 1: An example from machine translation.

Background: sequential vs. parallel framework

- RNNs and LSTMs deal with interaction by keeping a memory/summary of previous tokens. The underlying framework is **sequential**.

Background: sequential vs. parallel framework

- RNNs and LSTMs deal with interaction by keeping a memory/summary of previous tokens. The underlying framework is **sequential**.
- The analytic framework for transformers assume **parallel** access to either all tokens (or at least all relevant tokens).

Paper summary

1. Transformer networks are universal approximators for continuous seq-to-seq functions on compact domain.

Paper summary

1. Transformer networks are universal approximators for continuous seq-to-seq functions on compact domain.
2. Self-attention layers can compute contextual mappings of input sequences.

Review of transformer blocks

A **transformer block** is a seq-to-seq function $\mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ composed of:

- an **attention** layer that exposes *pairwise interaction*
- a **feedforward** layer to *perform computation*

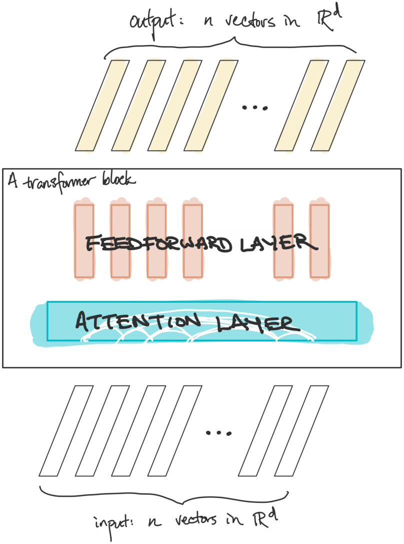
Review of transformer blocks

A **transformer block** is a seq-to-seq function $\mathbb{R}^{d\times n} \to \mathbb{R}^{d\times n}$ composed of:

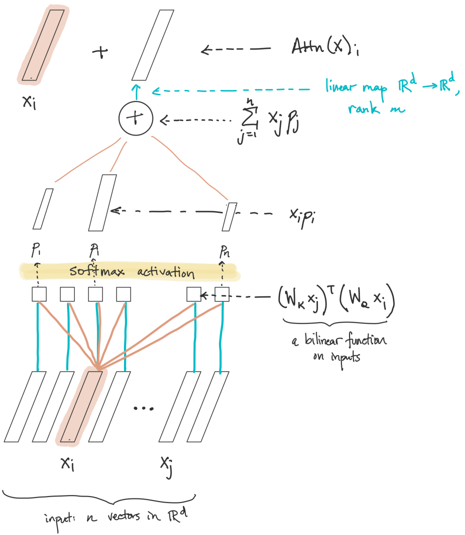
- an **attention** layer that exposes *pairwise interaction*
- a **feedforward** layer to *perform computation*

A **transformer network** is a composition of transformer blocks.

Review of transformer blocks



Review of transformer blocks: attention layer



Review of transformer blocks: attention layer

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$. Let σ be a column-wise softmax. Define:

$$\mathrm{Attn}(\mathbf{X}) = \mathbf{X} + \sum_{\ell=1}^{h} \mathbf{W}_O^\ell \mathbf{W}_V^\ell \mathbf{X} \cdot \sigma\left[(\mathbf{W}_K^\ell \mathbf{X})^\mathsf{T} (\mathbf{W}_Q^\ell \mathbf{X})\right]$$

Where $\mathbf{W}_K^\ell, \mathbf{W}_Q^\ell \in \mathbb{R}^{m \times d}$, $\mathbf{W}_O^\ell \in \mathbb{R}^{d \times m}$ and $\mathbf{W}_V^\ell \in \mathbb{R}^{m \times d}$.

Review of transformer blocks: attention layer

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$. Let σ be a column-wise softmax. Define:

$$\mathrm{Attn}(\mathbf{X}) = \mathbf{X} + \sum_{\ell=1}^{h} \mathbf{W}_O^\ell \mathbf{W}_V^\ell \mathbf{X} \cdot \sigma\left[(\mathbf{W}_K^\ell \mathbf{X})^\mathsf{T} (\mathbf{W}_Q^\ell \mathbf{X}) \right]$$

Where $\mathbf{W}_K^\ell, \mathbf{W}_Q^\ell \in \mathbb{R}^{m \times d}$, $\mathbf{W}_O^\ell \in \mathbb{R}^{d \times m}$ and $\mathbf{W}_V^\ell \in \mathbb{R}^{m \times d}$.

Review of transformer blocks: attention layer

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$. Let σ be a column-wise softmax. Define:

$$\mathrm{Attn}(\mathbf{X}) = \mathbf{X} + \sum_{\ell=1}^{h} \mathbf{W}_O^\ell \mathbf{W}_V^\ell \mathbf{X} \cdot \sigma\left[\, L^\ell(X \otimes X) \,\right]$$

Where L^ℓ is linear, $\mathbf{W}_O^\ell \in \mathbb{R}^{d \times m}$ and $\mathbf{W}_V^\ell \in \mathbb{R}^{m \times d}$.

Review of transformer blocks: attention layer

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$. Let σ be a column-wise softmax. Define:

$$\mathrm{Attn}(\mathbf{X}) = \mathbf{X} + \sum_{\ell=1}^{h} \mathbf{W}_O^\ell \mathbf{W}_V^\ell \mathbf{X} \cdot \sigma\left[L^\ell(X \otimes X)\right]$$

Where L^ℓ is linear, $\mathbf{W}_O^\ell \in \mathbb{R}^{d \times m}$ and $\mathbf{W}_V^\ell \in \mathbb{R}^{m \times d}$.

Review of transformer blocks: attention layer

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$. Let σ be a column-wise softmax. Define:

$$\mathrm{Attn}(\mathbf{X}) = \mathbf{X} + \sum_{\ell=1}^{h} T^{\ell} \left(\mathbf{X} \cdot \sigma \left[L^{\ell}(X \otimes X) \right] \right)$$

Where L^{ℓ} is linear, T^{ℓ} is linear and at most rank-m.

Review of transformer blocks: attention layer

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$. Let σ be a column-wise softmax. Define:

$$\mathrm{Attn}(\mathbf{X}) = \mathbf{X} + \sum_{\ell=1}^{h} T^{\ell}\left(\mathbf{X} \cdot \sigma\left[L^{\ell}(X \otimes X)\right]\right)$$

Where L^{ℓ} is linear, T^{ℓ} is linear and at most rank-m.

Review of transformer blocks: feedforward layer

Notice that the **feedforward layer** is just a single layer ReLu neural network applied componentwise; thus, it uses shared weights:

$$\mathrm{FF}(\mathbf{X}) = \mathbf{X} + \mathbf{W}_2 \cdot \mathrm{ReLu}\big(\mathbf{W}_1 \cdot \mathbf{X} + \mathbf{b}_1 \mathbf{1}_n^\mathsf{T}\big) + \mathbf{b}_2 \mathbf{1}_n^\mathsf{T}$$

for all tokens $\mathbf{X}_i$.

Review of transformer blocks: skip connections

Because of the **skip connections** in both the attention and feedforward layers, each block can turn on/off either layer:

$$\begin{aligned} \mathrm{Attn}(\mathbf{X}) &= \mathbf{X} + \sum_{\ell=1}^{h} \mathbf{W}_O^\ell \mathbf{W}_V^\ell \mathbf{X} \cdot \sigma\big[(\mathbf{W}_K^\ell \mathbf{X})^\mathsf{T} (\mathbf{W}_Q^\ell \mathbf{X})\big] \\ \mathrm{FF}(\mathbf{X}) &= \mathbf{X} + \mathbf{W}_2 \cdot \mathrm{ReLu}(\mathbf{W}_1 \cdot \mathbf{X} + \mathbf{b}_1 \mathbf{1}_n^\mathsf{T}) + \mathbf{b}_2 \mathbf{1}_n^\mathsf{T}, \end{aligned}$$

by setting $\mathbf{W}_O^\ell$ or $\big[\mathbf{W}_2\ \mathbf{b}_2\big]$ to zero.

Universal approximation intuition

1. An attention layer allows us to capture *pairwise interaction*. Compose attention layers for higher-order interaction.

Universal approximation intuition

1. An attention layer allows us to capture *pairwise interaction*. Compose attention layers for higher-order interaction.
2. Transform $\mathbf{X}$ into $\widetilde{\mathbf{X}}$, so that $\mathbf{X}$ is **encoded** into each $\widetilde{\mathbf{X}}_i$.

Universal approximation intuition

1. An attention layer allows us to capture *pairwise interaction*. Compose attention layers for higher-order interaction.
2. Transform $\mathbf{X}$ into $\widetilde{\mathbf{X}}$, so that $\mathbf{X}$ is **encoded** into each $\widetilde{\mathbf{X}}_i$.
   - We'll call this a *contextual mapping* as $\widetilde{\mathbf{X}}_i$ captures not just $\mathbf{X}_i$ but its context $\mathbf{X}$.

Universal approximation intuition

1. An attention layer allows us to capture *pairwise interaction*. Compose attention layers for higher-order interaction.
2. Transform $\mathbf{X}$ into $\widetilde{\mathbf{X}}$, so that $\mathbf{X}$ is **encoded** into each $\widetilde{\mathbf{X}}_i$.
    - We'll call this a *contextual mapping* as $\widetilde{\mathbf{X}}_i$ captures not just $\mathbf{X}_i$ but its context $\mathbf{X}$.
3. Apply usual universal approximation of feedforward neural nets to get approximation result.

Caveat: permutation equivariance

Definition

A map $f : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ *is* **permutation equivariant** *if for all permutations* $P \in \Sigma_n$ *and input* $\mathbf{X} \in \mathbb{R}^{d \times n}$*,*

$$f(\mathbf{X}P) = f(\mathbf{X})P.$$

Caveat: permutation equivariance

Claim

Transformer blocks are permutation equivariant.

Caveat: permutation equivariance

Claim

Transformer blocks are permutation equivariant.

Proof.

The proof is direct. Let P be a permutation matrix. Then:

$$
\begin{aligned}
\mathrm{Attn}(\mathbf{X}P) &= \mathbf{X}P + \sum_{\ell=1}^{h} \mathbf{W}^{\ell}_{O,V}\mathbf{X}P \cdot \sigma\big[(\mathbf{W}^{\ell}_{K}\mathbf{X}P)^{\mathsf{T}}(\mathbf{W}^{\ell}_{Q}\mathbf{X}P)\big] \\
&= \mathbf{X}P + \sum_{\ell=1}^{h} \mathbf{W}^{\ell}_{O,V}\mathbf{X}PP^{\mathsf{T}} \cdot \sigma\big[(\mathbf{W}^{\ell}_{K}X)^{\mathsf{T}}(\mathbf{W}^{\ell}_{Q}\mathbf{X})\big]P \\
&= \mathrm{Attn}(\mathbf{X})P,
\end{aligned}
$$

since $PP^{\mathsf{T}} = I$. Proof similar for $\mathrm{FF}(\mathbf{X}P) = \mathrm{FF}(\mathbf{X})P$. □

Main result

Universal approximation theorem

Theorem

Let $f : \mathbb{R}^{d\times n} \to \mathbb{R}^{d\times n}$ *be a continuous and permutation equivariant function with compact support.*

Universal approximation theorem

Theorem

Let $f : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ *be a continuous and permutation equivariant function with compact support. Then, for all* $1 \leq p < \infty$ *and* $\epsilon > 0$, *there is a transformer network* g *such that:*

$$\mathsf{d}_p(f, g) := \left(\int \| f(\mathbf{X}) - g(\mathbf{X}) \|_p^p \, d\mathbf{X} \right)^{1/p} < \epsilon.$$

Generalized transformers

We will consider **generalized transformers** where:

Generalized transformers

We will consider **generalized transformers** where:

- We may replace the softmax σ by a hardmax σ_H.

Generalized transformers

We will consider **generalized transformers** where:

- We may replace the softmax σ by a hardmax σ_H.
- We may replace ReLu with any $\phi \in \Phi$, where Φ consists of 3-piecewise linear function with at least one constant piece.

Approximation of generalized transformers

Lemma

The class of transformers is dense in the class of generalized transformers (with respect to the metric d_p*).*

Approximation of generalized transformers

Lemma

The class of transformers is dense in the class of generalized transformers (with respect to the metric d_p*).*

Proof sketch.

- $\sigma(\lambda \mathbf{A}) \to \sigma_H(\mathbf{A})$ as $\lambda \to \infty$.
- Any $\phi \in \Phi$ can be arbitrarily approximated by four ReLu's.

$\square$

Proof sketch

1. Without loss of generality, let $\mathrm{supp}(f) = [0, 1]^{d \times n}$.

Proof sketch

1. Without loss of generality, let $\mathrm{supp}(f) = [0,1]^{d\times n}$.
2. Quantize $[0,1]^d$ to a grid $\mathbb{G}_\delta := \{0, \delta, \ldots, 1-\delta\}^{d\times n}$.

Proof sketch

1. Without loss of generality, let $\mathrm{supp}(f) = [0,1]^{d\times n}$.
2. Quantize $[0,1]^d$ to a grid $\mathbb{G}_\delta := \{0, \delta, \ldots, 1-\delta\}^{d\times n}$.
3. Prove transformers can quantize the cube, $[0,1]^{d\times n} \to \mathbb{G}_\delta$.

Proof sketch

1. Without loss of generality, let $\mathrm{supp}(f) = [0,1]^{d\times n}$.
2. Quantize $[0,1]^d$ to a grid $\mathbb{G}_\delta := \{0, \delta, \ldots, 1-\delta\}^{d\times n}$.
3. Prove transformers can quantize the cube, $[0,1]^{d\times n} \to \mathbb{G}_\delta$.
4. Prove approximation result for functions $\mathbb{G}_\delta \to \mathbb{R}^{d\times n}$.

Quantization

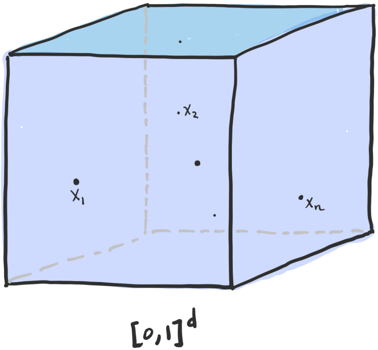


Figure 2: An input $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n) \in \mathbb{R}^{d \times n}$

Quantization

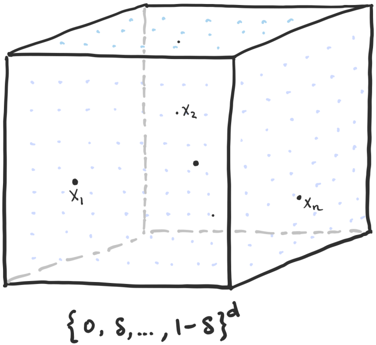


Figure 2: An input $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n) \in \mathbb{R}^{d \times n}$

Quantization

Define the **scalar quantization map** $g_{\text{s.q.}}$ as:

$$g_{\text{s.q.}}(x) = \begin{cases} \lfloor x/\delta \rfloor \cdot \delta & x \in [0,1) \\ -\delta^{nd} & \text{o.w.} \end{cases}$$

Quantization

Define the **scalar quantization map** $g_{\text{s.q.}}$ as:

$$g_{\text{s.q.}}(x) = \begin{cases} \lfloor x/\delta \rfloor \cdot \delta & x \in [0,1) \\ -\delta^{nd} & \text{o.w.} \end{cases}$$

The map $g_{\text{s.q.}}$ rounds $x \in [0,1)$ down to the nearest grid point.

Feedforward layers can quantize the cube

Lemma

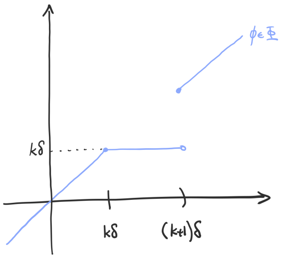
The map $q_{\mathrm{s.q.}}$ *can be implemented using* $\left(\frac{1}{\delta}+1\right)$ *feedforward layers with activation* $\phi \in \Phi$ *and each hidden layer dimension equal to 1.*

Feedforward layers can quantize the cube

Lemma

The map $q_{\mathrm{s.q.}}$ can be implemented using $\left(\frac{1}{\delta}+1\right)$ feedforward layers with activation $\phi \in \Phi$ and each hidden layer dimension equal to 1.

Proof sketch.



More generally, $[0,1]^d$ can be quantized using $d\left(\frac{1}{\delta}+1\right)$ layers. $\square$

Contextual mapping

Definition

Let $\mathbb{L} \subset \mathbb{R}^{d \times n}$ *be a finite set. Let* $L, L' \in \mathbb{L}$*. A* **contextual mapping** *is a map* $q : \mathbb{L} \to \mathbb{R}^{1 \times n}$ *such that*

Contextual mapping

Definition

Let $\mathbb{L} \subset \mathbb{R}^{d \times n}$ *be a finite set. Let* $L, L' \in \mathbb{L}$*. A* **contextual mapping** *is a map* $q : \mathbb{L} \to \mathbb{R}^{1 \times n}$ *such that*

- *If two columns are distinct* $L_i \neq L_j$*, then* $q(L)_i \neq q(L)_j$*.*

Contextual mapping

Definition

Let $\mathbb{L} \subset \mathbb{R}^{d\times n}$ be a finite set. Let $L, L' \in \mathbb{L}$. A **contextual mapping** *is a map $q : \mathbb{L} \to \mathbb{R}^{1\times n}$ such that*

- *If two columns are distinct $L_i \neq L_j$, then $q(L)_i \neq q(L)_j$.*
- *If L and L' are not permutations, then for $1 \leq i, i' \leq n$:*

$$q(L)_i \neq q(L')_{i'}.$$

Contextual mapping

Definition

Let $\mathbb{L} \subset \mathbb{R}^{d \times n}$ be a finite set. Let $L, L' \in \mathbb{L}$. A **contextual mapping** *is a map $q : \mathbb{L} \to \mathbb{R}^{1 \times n}$ such that*

- *If two columns are distinct $L_i \neq L_j$, then $q(L)_i \neq q(L)_j$.*
- *If L and L' are not permutations, then for $1 \leq i, i' \leq n$:*

$$q(L)_i \neq q(L')_{i'}.$$

In other words, if $q(L)_i$ is an encoding of L_i, then from $q(L)_i$, we can recover not only L_i but also its context L.

Contextual mapping

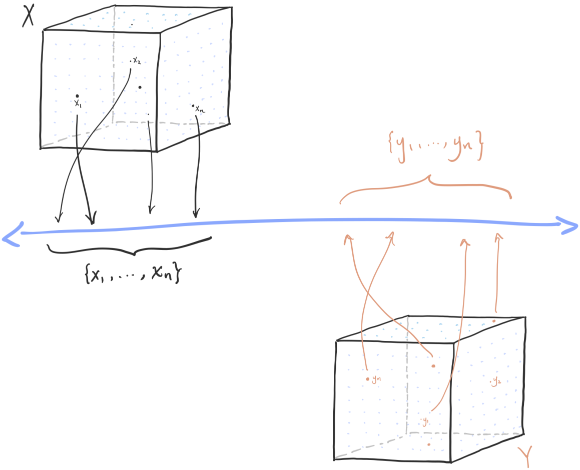


Figure 3: A contextual mapping $q : \mathbb{G}_\delta \to \mathbb{R}^{1\times n}$.

Simplification of $\mathbb{G}_\delta$

Define $\widetilde{\mathbb{G}}_\delta \subset \mathbb{G}_\delta$ by:

$$\widetilde{\mathbb{G}}_\delta := \{L \in \mathbb{G}_\delta : L \text{ has distinct columns}\}.$$

Simplification of $\mathbb{G}_\delta$

Define $\widetilde{\mathbb{G}}_\delta \subset \mathbb{G}_\delta$ by:

$$\widetilde{\mathbb{G}}_\delta := \{L \in \mathbb{G}_\delta : L \text{ has distinct columns}\}.$$

- Notice that $\left|\mathbb{G}_\delta - \widetilde{\mathbb{G}}_\delta\right| = O\left(\delta^d |\mathbb{G}_\delta|\right)$, so $\widetilde{\mathbb{G}}_\delta$ contain essentially all of $\mathbb{G}_\delta$.

Self-attention layers can express contextual mappings

Lemma

There exists a map $g_{\mathrm{con}} : \mathbb{R}^{d \times n} \to \mathbb{R}^{d \times n}$ *composed of* $\frac{1}{\delta^d} + 1$ *self-attention layers, vector* $u \in \mathbb{R}^d$ *and* $q(L) := u^\top g_{\mathrm{con}}(L)$ *where:*

1. q *is a contextual mapping of* $\widetilde{\mathbb{G}}_\delta$
2. *There exists* $0 < a < b$ *such that:*
   a. *if* $L \in \widetilde{\mathbb{G}}_\delta$*, then* $q(L) \in [a, b]^{1 \times n}$
   b. *if* $L \in \{-\delta^{nd}, 0, \delta, \ldots, 1 - \delta\}^{d \times n} - \widetilde{\mathbb{G}}_\delta$*, then* $q(L) \notin [a, b]^{1 \times n}$.

Feedforward layers can interpolate

Lemma

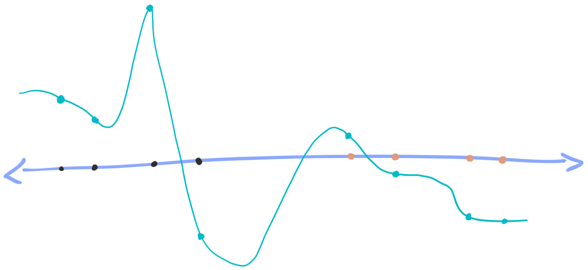
Let $f : \widetilde{\mathbb{G}}_\delta \to \mathbb{R}^{d\times n}$ be any permutation equivariant function. Given g_{con} as above, there exists $g_{\mathrm{out}} : \mathbb{R}^{d\times n} \to \mathbb{R}^{d\times n}$ a composition of feedforward layers such that:

$$g_{\mathrm{out}} \circ g_{\mathrm{con}}(L) = \begin{cases} f(L) & L \in \widetilde{\mathbb{G}}_\delta \\ \mathbf{0} & L \in \{-\delta^{nd}, 0, \delta, \dots, 1-\delta\}^{d\times n} - \widetilde{\mathbb{G}}_\delta. \end{cases}$$

Proof: feedforward layers can interpolate

Proof sketch.

A neural network can interpolate on $O\left(n\delta^{-dn}/n!\right)$ points of $\mathbb{R}$.



We need at most $O(n\delta^{-dn}/n!)$ layers with hidden dimension 1. $\square$

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

- WLOG, $\widetilde{G}_\delta \ni L = \frac{1}{N}(k_1, \ldots, k_n)$ where $k_i \in \mathbb{N}$.

$\square$

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

- WLOG, $\widetilde{G}_\delta \ni L = \frac{1}{N}(k_1, \dots, k_n)$ where $k_i \in \mathbb{N}$.
- Attention can compute $r = \max_{i,j} k_i - k_j$ and add Nr to any coordinate i such that $k_i = k$.

$\square$

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

- WLOG, $\widetilde{G}_\delta \ni L = \frac{1}{N}(k_1, \ldots, k_n)$ where $k_i \in \mathbb{N}$.
- Attention can compute $r = \max_{i,j} k_i - k_j$ and add Nr to any coordinate i such that $k_i = k$.
- Sequentially apply transform, for $k = 0, \ldots, N - 1$.

$\square$

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

- WLOG, $\widetilde{G}_\delta \ni L = \frac{1}{N}(k_1, \ldots, k_n)$ where $k_i \in \mathbb{N}$.
- Attention can compute $r = \max_{i,j} k_i - k_j$ and add Nr to any coordinate i such that $k_i = k$.
- Sequentially apply transform, for $k = 0, \ldots, N - 1$.
  - If O is the output, then $k_i/N = O_i \mod 1$.

$\square$

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

- WLOG, $\widetilde{G}_\delta \ni L = \frac{1}{N}(k_1, \ldots, k_n)$ where $k_i \in \mathbb{N}$.
- Attention can compute $r = \max_{i,j} k_i - k_j$ and add Nr to any coordinate i such that $k_i = k$.
- Sequentially apply transform, for $k = 0, \ldots, N-1$.
  - If O is the output, then $k_i/N = O_i \mod 1$.
  - All k_i's can be recovered from $\max_j O_j$.

$\square$

Proof: self-attention can express contextual mappings

Proof sketch, 1D case.

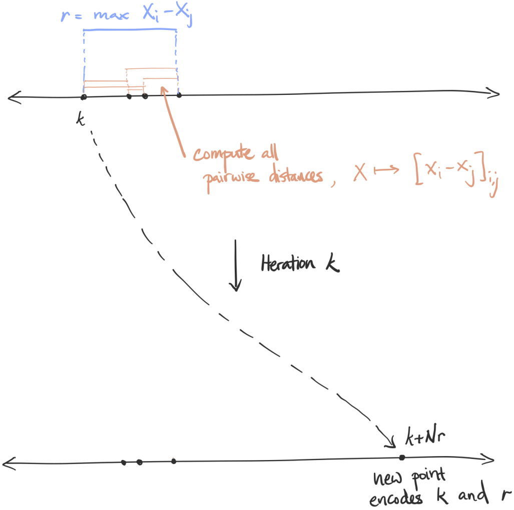
Let $d = 1$. We can use $\frac{1}{\delta} + 1$ attention layers to construct a contextual mapping of $\widetilde{\mathbb{G}}_\delta$. Assume $\frac{1}{\delta} = N \in \mathbb{N}$.

- WLOG, $\widetilde{G}_\delta \ni L = \frac{1}{N}(k_1, \ldots, k_n)$ where $k_i \in \mathbb{N}$.
- Attention can compute $r = \max_{i,j} k_i - k_j$ and add Nr to any coordinate i such that $k_i = k$.
- Sequentially apply transform, for $k = 0, \ldots, N - 1$.
  - If O is the output, then $k_i/N = O_i \mod 1$.
  - All k_i's can be recovered from $\max_j O_j$.
- Attention can compute $s = \max_j O_j$ and add $N^{n+1}s$ to each coordinate. Thus, all k_i's are encoded into $O_j + N^{n+1}s$.

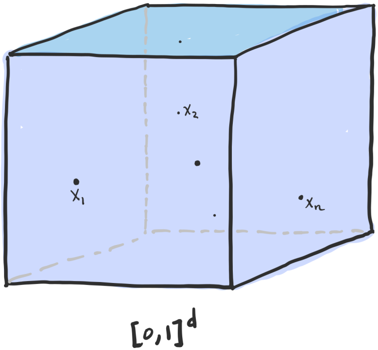
□

Proof: self-attention can express contextual mappings

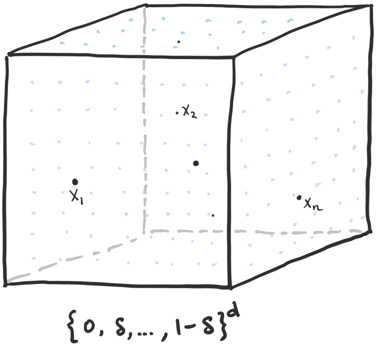
For $k = 0, \dots, N-1$:



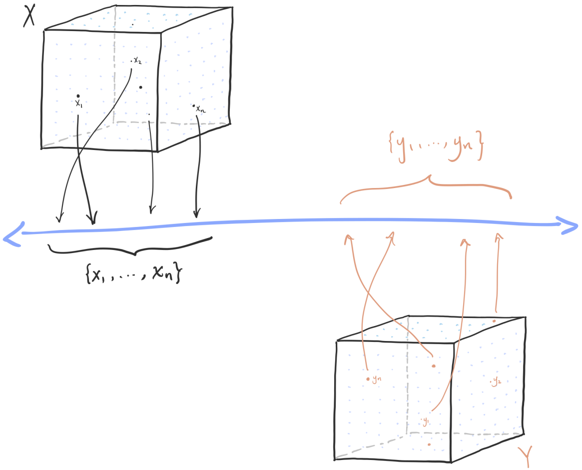
A summary in pictures



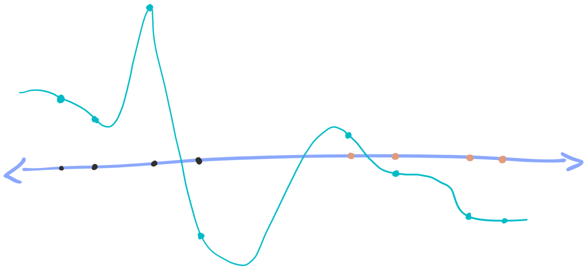
A summary in pictures



A summary in pictures



A summary in pictures



Position encoding

If we can preprocess $\mathbf{X} \in [0,1]^{d \times n}$ via **position encoding**:

$$\mathbf{X} \mapsto \mathbf{X} + \mathbf{1} \cdot (0, \ldots, n-1),$$

then the universal approximation theorem extends to general sequence-to-sequence functions.