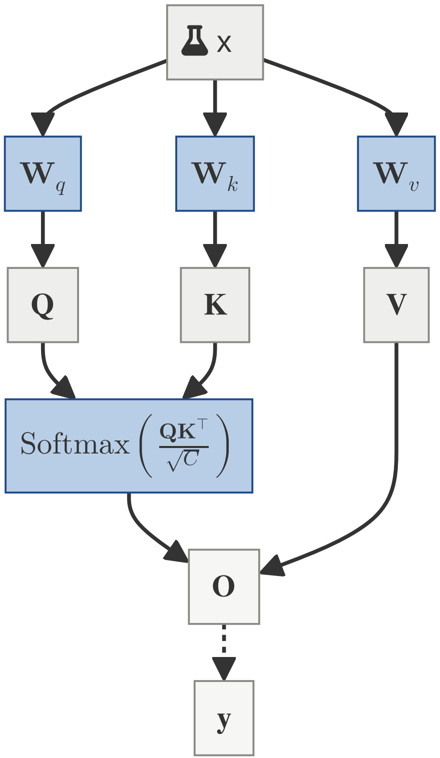
Positional Embeddings

Recap: Attention (With Weights)

$$
\begin{aligned}
&\text{Attention}(\mathbf{X}; \mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V) \\
&\quad = \text{Attention}(\mathbf{X}\mathbf{W}_Q, \mathbf{X}\mathbf{W}_K, \mathbf{X}\mathbf{W}_V) \\
&\quad = \text{Softmax}\left(\frac{\mathbf{X}\mathbf{W}_Q(\mathbf{X}\mathbf{W}_K)^\top}{\sqrt{d_k}}\right)\mathbf{X}\mathbf{W}_V
\end{aligned}
$$



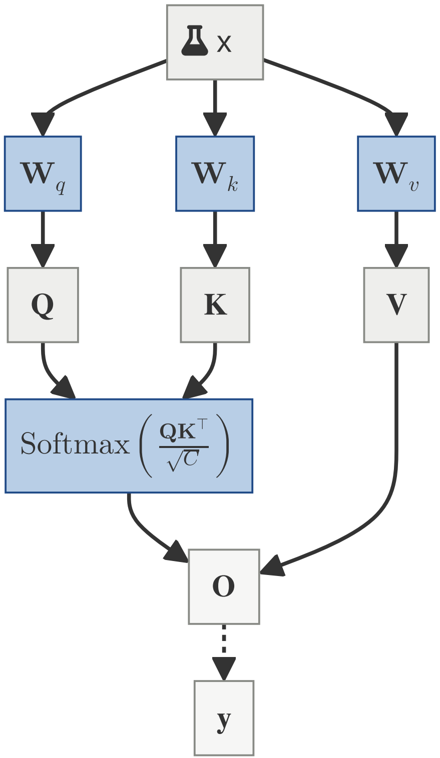
Permutation Invariance

Attention is a *set* operation

- shuffling keys/values gives the same output

Let $\mathrm{Perm}(\mathbf{M})$ denote an arbitary permutation operation over the rows of a matrix $\mathbf{M}$

$$\begin{aligned}\mathrm{Attention}(\mathbf{X}\mathbf{W}_Q, \mathrm{Perm}(\mathbf{X}\mathbf{W}_K), \mathrm{Perm}(\mathbf{X}\mathbf{W}_V)) \\ = \mathrm{Attention}(\mathbf{X}\mathbf{W}_Q, \mathbf{X}\mathbf{W}_K, \mathbf{X}\mathbf{W}_V)\end{aligned}$$



Example: Set Operations

Given a set of reported power outages in the last few days, predict the number of power outages in the future

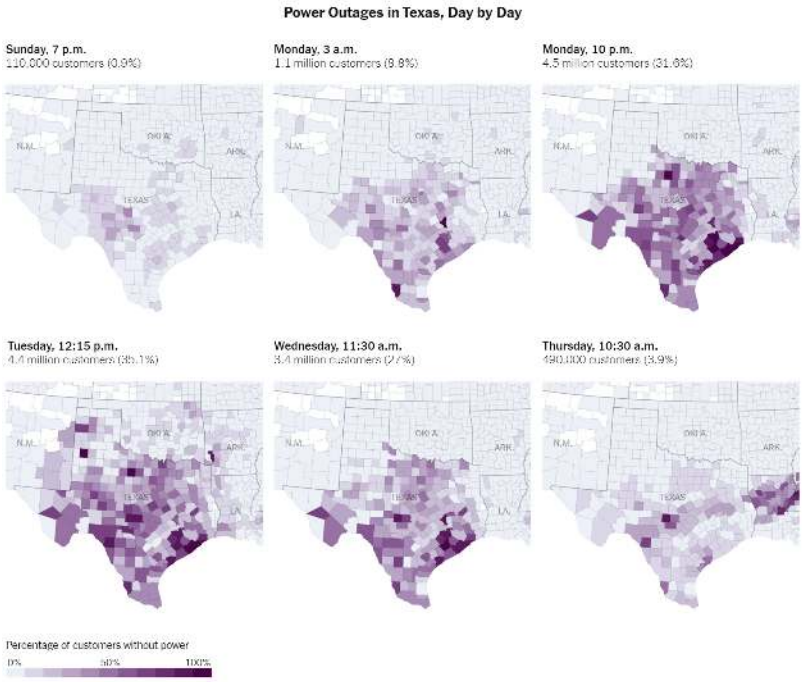


Image credit: New York Times ↩️

Example: Ordered Operations

Natural Language - ordered array of words

`my kid likes the movie ≠ the movie likes my kid`

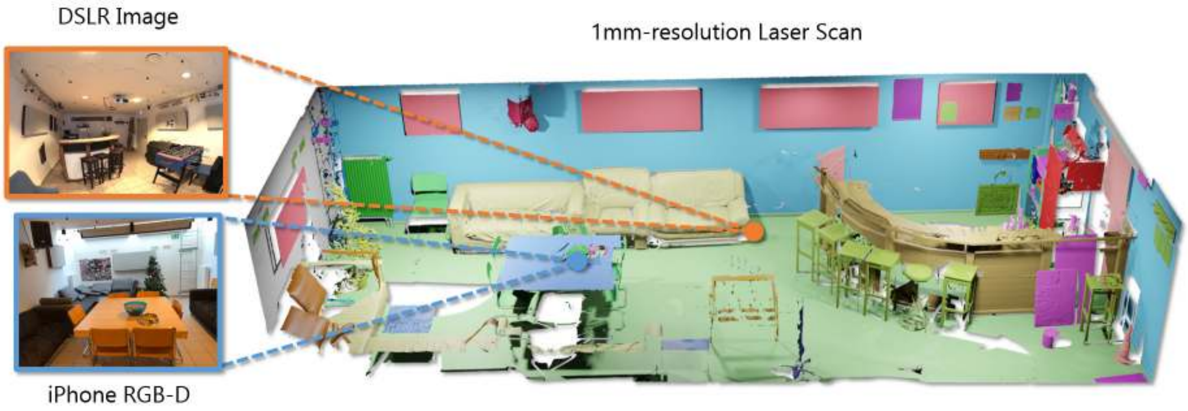
Speech - sequence of sound waves

Images - ordered set of smaller patches

- What happens when you shuffle an image?
- Fun demo: Visual Anagrams[1]

1. Geng, *et al.* Visual Anagrams: Generating Multi-View Optical Illusions with Diffusion Models. CVPR 2024 ↩️

Example: Somewhat Ordered Operations



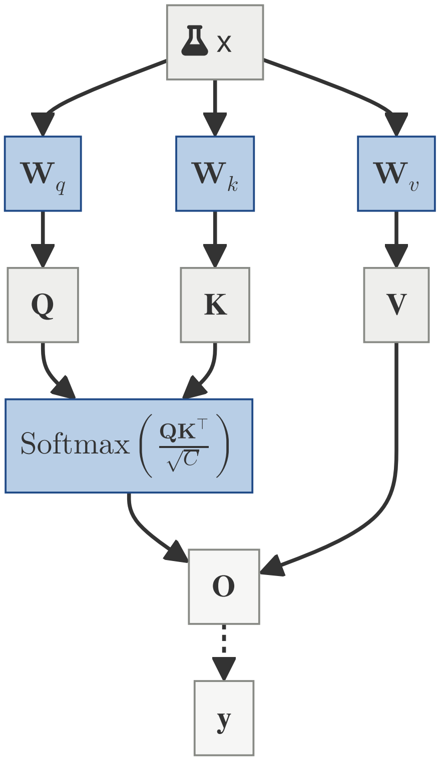
Point Clouds

- Set of N points $p_i \in \mathbb{R}^3$
- $P = \{p_1, p_2, \ldots, p_N\}$

Yeshwanth, *et al.* ScanNet++: A High-Fidelity Dataset of 3D Indoor Scenes. ICCV 2023 ↩️

Attention Without Positional Embedding

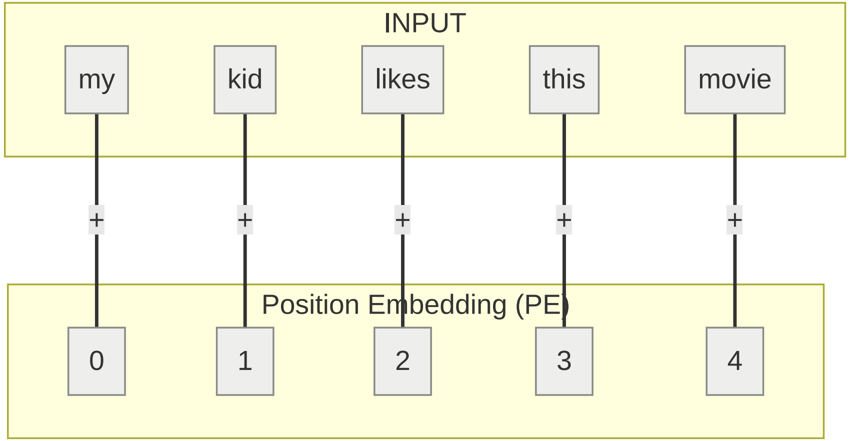
$$\begin{aligned}\text{Attention}(\mathbf{X}; \mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V) \\ &= \text{Attention}(\mathbf{X}\mathbf{W}_Q, \mathbf{X}\mathbf{W}_K, \mathbf{X}\mathbf{W}_V) \\ &= \text{Softmax}\left(\frac{\mathbf{X}\mathbf{W}_Q(\mathbf{X}\mathbf{W}_K)^\top}{\sqrt{d_k}}\right)\mathbf{X}\mathbf{W}_V\end{aligned}$$

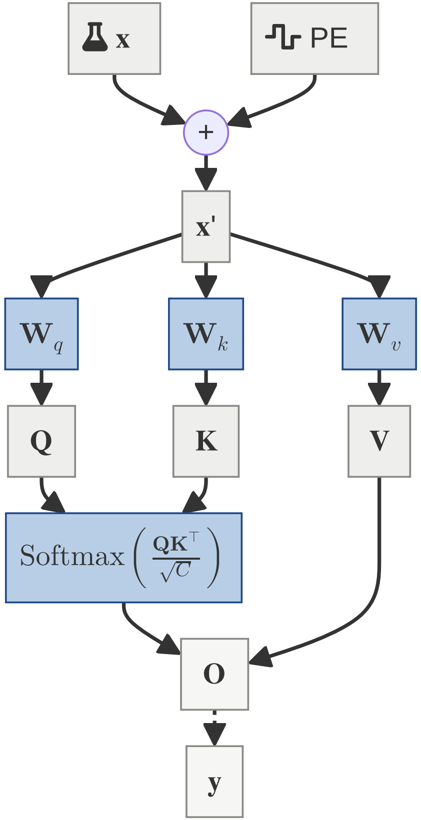


Attention With Positional Embedding

Describes the location of elements in a sequence

- add position information to $\mathbf{Q}, \mathbf{K}, \mathbf{V}$



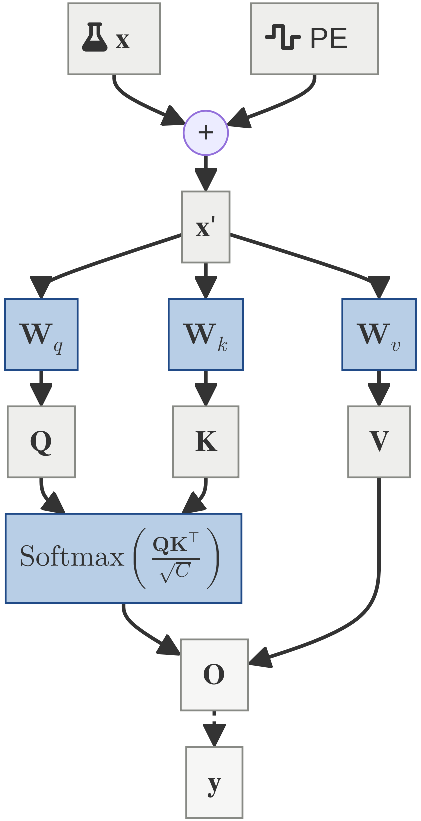


Absolute Positional Embeddings

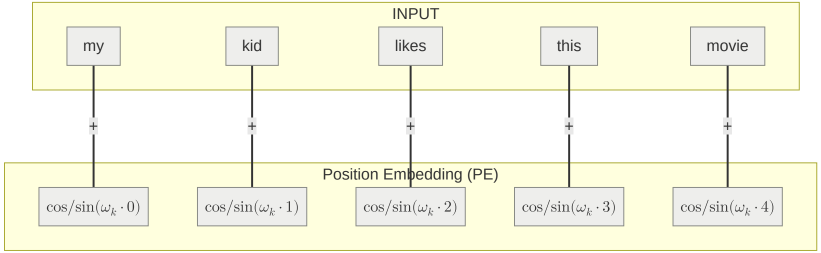
Option 1: use the absolute, raw position

✓ Straightforward

✗ Not meaningful



Sinusoidal Positional Embeddings



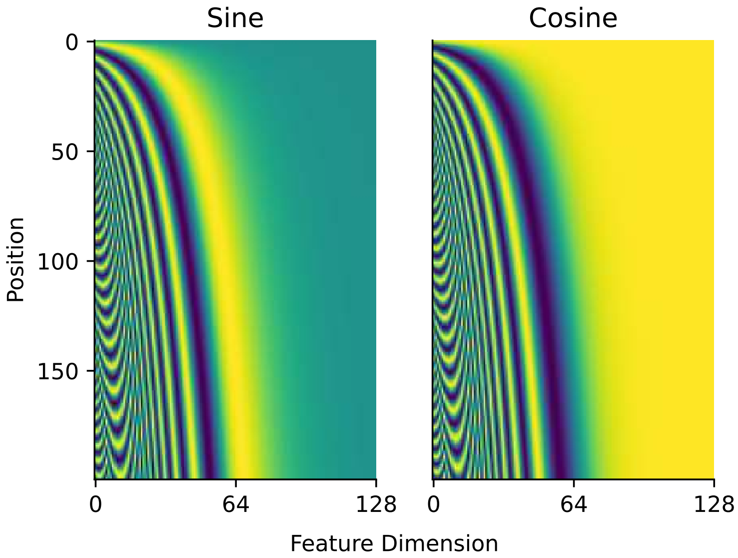
Option 2: encode position with sine/cosine

- Use sine & cosine functions with varied frequencies

$$\mathrm{PE} \in \mathbb{R}^{N \times C}$$

$$\mathrm{PE}(n, 2i) = \sin\left(\frac{n}{10000^{2i/C}}\right)$$

$$\mathrm{PE}(n, 2i+1) = \cos\left(\frac{n}{10000^{2i/C}}\right)$$

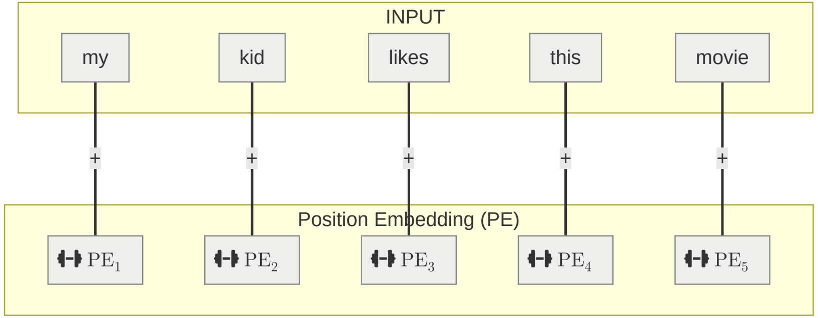


- ✓ "Kind of" absolute
- ✓ Position represented well by frequency/phase

Learnable Positional Embedding

Option 3: learned encoding

- Embeddings $\mathrm{PE} \in \mathbb{R}^{N \times C}$ randomly initialized
- Learned through training

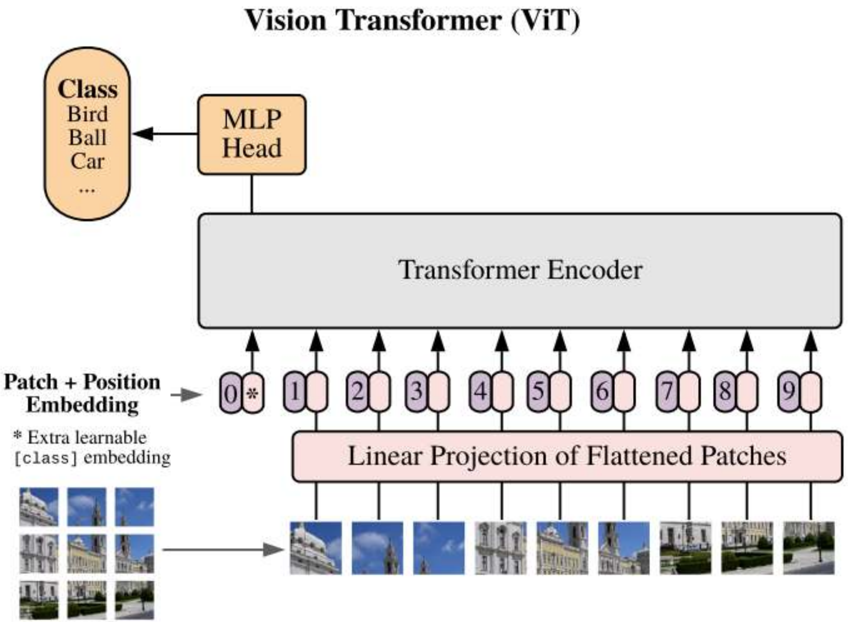


✓ Fully learned

✓ Use when frequency information is not obvious

✗ Performance drops when the sequence length varies between train/test

Learnable Positional Embedding



Dosovitskiy, *et al.* "An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021 ↩️

Relative Positional Embedding (1): T5-bias[1]:

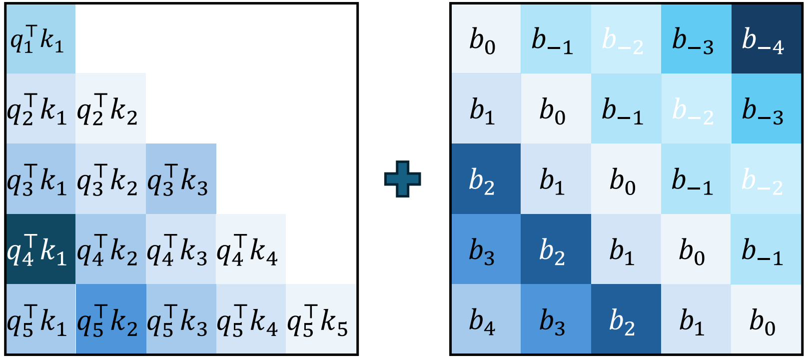
Option 4a: pairwise/relative encoding

- Takes the form of $\mathrm{PE}(m, n) = f(m, n)$

$$\mathbf{O} = \mathrm{softmax}\left(\frac{\mathbf{XW}_Q(\mathbf{XW}_K)^\top + \mathbf{B}}{\sqrt{C}}\right)(\mathbf{XW}_V)$$

$$B_{ij} = b_{i-j}$$

$$\mathbf{B} = \begin{bmatrix} b_0 & b_{-1} & b_{-2} & \cdots & b_{-N+1} \\ b_1 & b_0 & b_{-1} & \cdots & b_{-N+2} \\ b_2 & b_1 & b_0 & \cdots & b_{-N+3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{N-1} & b_{N-2} & b_{N-3} & \cdots & b_0 \end{bmatrix}$$



✓ Generalizes better to sequences of unseen lengths

1. Raffel, *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR 2020. ↩

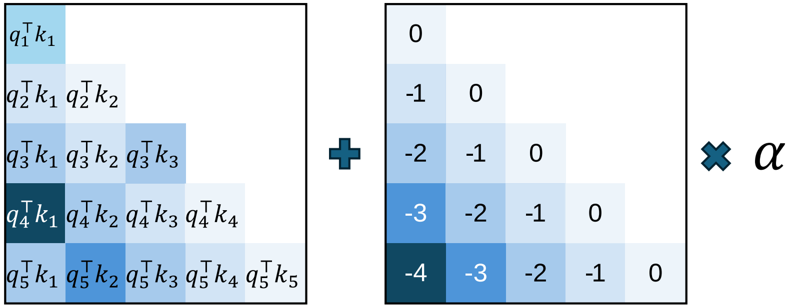
Relative Positional Embedding (2): Alibi[1]:

Option 4b: pairwise/relative encoding

- $\mathrm{PE}(m, n)$ depends on the pair of positions (m, n)

$$\mathbf{O} = \mathrm{softmax}\left(\frac{\mathbf{XW}_Q(\mathbf{XW}_K)^\top + \mathbf{B}}{\sqrt{C}}\right)(\mathbf{XW}_V)$$

$$\mathbf{B} = \alpha \cdot \begin{bmatrix} 0 & \cdots & & & \\ -1 & 0 & \cdots & & \\ -2 & -1 & 0 & \cdots & \\ \vdots & \vdots & \vdots & \ddots & \\ -N+1 & -N+2 & -N+3 & \cdots & 0 \end{bmatrix}$$



✓ Generalizes better to sequences of unseen lengths

1. Press *et al.* Train short, test long: Attention with linear biases enables input length extrapolation. ICLR 2022 ↩

Relative Positional Embedding (3)

Option 4c: pairwise/relative encoding

- Encode edge between two arbitary positions i and j

$$e_{ij} = \left(\frac{\mathbf{x}_i \mathbf{W}_Q (\mathbf{x}_j \mathbf{W}_K + \mathbf{P}_{ij}^K)^\top}{\sqrt{C}} \right)$$

$$\mathbf{o}_i = \sum_{j=1}^{N} \alpha_{ij} (\mathbf{x}_j \mathbf{W}_V + \mathbf{P}_{ij}^V)$$

Shaw *et al.* Self-Attention with Relative Position Representations. NAACL 2018. ↩

Rotary Positional Embedding: RoPE[1]:

Option 5: rotary encoding

- Both absolute PE and relative PE
- Goal: find a kernel function such that

$$h(\mathbf{q}_m, \mathbf{k}_n) = \mathbf{q}_m^\top \mathbf{k}_n = g(\mathbf{q}_m, \mathbf{k}_n, m - n)$$

$$\mathbf{q}_m = \mathbf{R}_m \mathbf{W}_Q \mathbf{x}_m \qquad \mathbf{k}_n = \mathbf{R}_n \mathbf{W}_K \mathbf{x}_n$$

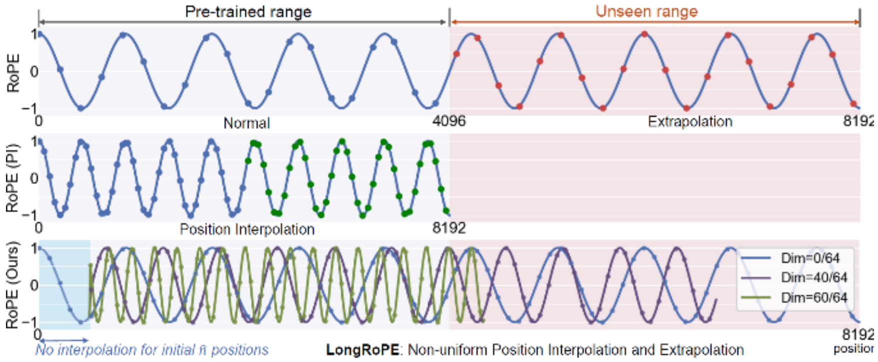
$$\text{where } \mathbf{R}_m = \begin{bmatrix} \cos(m\theta_1) & -\sin(m\theta_1) & 0 & 0 & \cdots & 0 & 0 \\ \sin(m\theta_1) & \cos(m\theta_1) & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos(m\theta_2) & -\sin(m\theta_2) & \cdots & 0 & 0 \\ 0 & 0 & \sin(m\theta_2) & \cos(m\theta_2) & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos(m\theta_{C/2}) & -\sin(m\theta_{C/2}) \\ 0 & 0 & 0 & 0 & \cdots & \sin(m\theta_{C/2}) & \cos(m\theta_{C/2}) \end{bmatrix}$$

1. Su *et al.* RoFormer: Enhanced Transformer with Rotary Position Embedding. Neurocomputing 2024. ↩

Rotary Positional Embedding

$$\mathbf{q}_m = \mathbf{R}_m \mathbf{W}_Q \mathbf{x}_m$$
$$\mathbf{k}_n = \mathbf{R}_n \mathbf{W}_K \mathbf{x}_n$$

$$\mathbf{q}_m^\top \mathbf{k}_n = (\mathbf{R}_m \mathbf{W}_Q \mathbf{x}_m)^\top \mathbf{R}_n \mathbf{W}_K \mathbf{x}_n = \mathbf{x}_m^\top \mathbf{W}_Q^\top \mathbf{R}_{n-m} \mathbf{W}_k \mathbf{x}_n$$



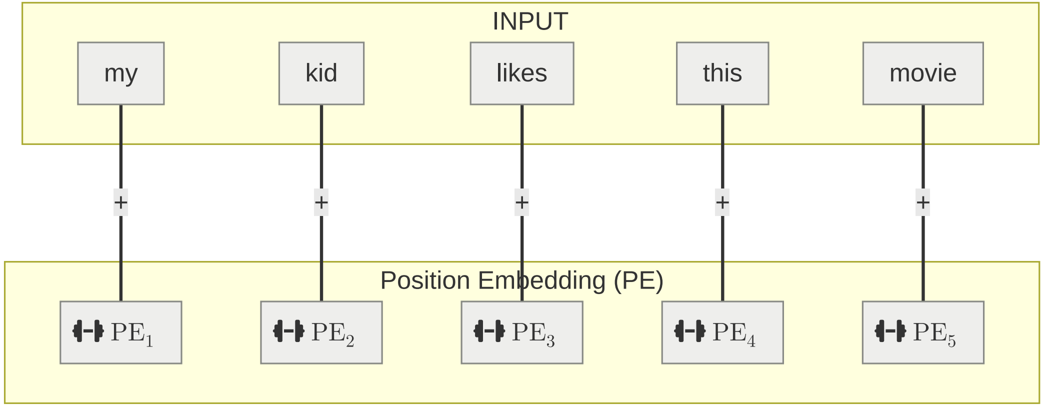
✓ Great extrapolation capability when context length between train/test varies

Widely adopted in Large Language Models (LLMs) such as LLaMA[1]:

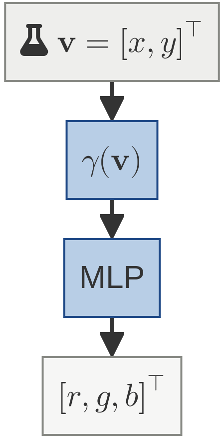
1. Touvron, *et al*. LLaMA: Open and efficient foundation language models. arXiv:2302.13971 ↩

Applications of PE: LLM

PEs are widely used in Large Language Models (LLM)



Applications of PE: Implicit Functions



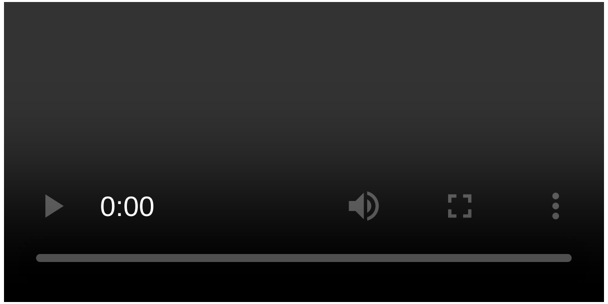
Implicit Functions

- $f : \mathbb{R}^2 \to \mathbb{R}^3$ modeled by a network (e.g. MLP)
- **Input**: pixel coordinate $\mathbf{v} = [x, y]^\top$
- **Output**: color value $[r, g, b]^\top$

Fourier feature mapping[1]:

$$\gamma(\mathbf{v}) = \begin{bmatrix} \cos(2\pi \mathbf{B}\mathbf{v}) \\ \sin(2\pi \mathbf{B}\mathbf{v}) \end{bmatrix}$$

$\mathbf{B}$ is a random Gaussian matrix: $\mathbf{B}_{i,j} \sim \mathcal{N}(0, \sigma^2)$



1. Tancik *et al.* Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. NeurIPS 2020. ↩

Positional Embeddings - TL;DR

Positional embeddings are used to break permutation invariance

Positional embeddings encode location-related information

Many types of PEs: absolute, sinusoidal, learnable, relative, rotary