Positional Embedding

Vijay Raghavan

September 2024

Summary

Sinusoidal embeddings use sine and cosine functions of different frequencies to create a unique encoding for each position in the sequence. This helps the transformer model keep track of the order of words, even though it processes all words in parallel. In our example, we calculated the positional encoding for the word at position 2 using the sine and cosine functions, resulting in a unique vector that the model uses to understand this word's position in the sequence.

Sinusoidal Positional Encoding: Example

Transformers use sinusoidal positional encodings to give the model a sense of the position of each word in a sequence. Since transformers process all words simultaneously, they need a way to understand the order of the words. Sinusoidal embeddings achieve this using sine and cosine functions with different frequencies.

The Sinusoidal Encoding Formulas

Given a position pos and dimension i (where i is an index into the embedding dimension):

$$\begin{aligned}&\text{For even indices } (2i):\\ &\quad \mathrm{PE}_{(\mathrm{pos},2i)} = \sin\left(\frac{\mathrm{pos}}{10000^{2i/d_{\mathrm{model}}}}\right)\\ &\text{For odd indices } (2i+1):\\ &\quad \mathrm{PE}_{(\mathrm{pos},2i+1)} = \cos\left(\frac{\mathrm{pos}}{10000^{2i/d_{\mathrm{model}}}}\right)\end{aligned}$$

Here, d_{model} is the dimension of the model.

Example: Sinusoidal Encoding in Action

Assume:

- We want to encode the position $\text{pos} = 2$.
- The model's embedding dimension $d_{\text{model}} = 4$.

Step 1: Calculate the Encoding for pos $= 2$

1. For $i = 0$ (the first dimension):

   $i = 0$ is even, so we use the sine formula:

   $$\text{PE}_{(2,0)} = \sin\left(\frac{2}{10000^{0/4}}\right) = \sin\left(\frac{2}{1}\right) = \sin(2)$$

   Assume $\sin(2) \approx 0.909$

2. For $i = 1$ (the second dimension):

   $i = 1$ is odd, so we use the cosine formula:

   $$\text{PE}_{(2,1)} = \cos\left(\frac{2}{10000^{2/4}}\right) = \cos\left(\frac{2}{100}\right) = \cos(0.02)$$

   Assume $\cos(0.02) \approx 0.9998$

3. For $i = 2$ (the third dimension):

   $i = 2$ is even, so we use the sine formula:

   $$\text{PE}_{(2,2)} = \sin\left(\frac{2}{10000^{4/4}}\right) = \sin\left(\frac{2}{10000}\right) = \sin(0.0002)$$

   Assume $\sin(0.0002) \approx 0.0002$

4. For $i = 3$ (the fourth dimension):

   $i = 3$ is odd, so we use the cosine formula:

   $$\text{PE}_{(2,3)} = \cos\left(\frac{2}{10000^{6/4}}\right) = \cos\left(\frac{2}{1000000}\right) = \cos(0.000002)$$

   Assume $\cos(0.000002) \approx 1.0$

Step 2: Combine the Encodings

The positional encoding for position $\text{pos} = 2$ with embedding dimension $d_{\text{model}} = 4$ is:

$$\text{PE}_2 = [0.909, 0.9998, 0.0002, 1.0]$$

How This Helps the Model

Each position in the input sequence gets a unique positional encoding vector like the one above. These vectors are added to the word embeddings of the input sequence, allowing the transformer to process the sequence with a sense of order. By using sine and cosine functions of different frequencies, the model can easily capture both absolute and relative positions of words in the sequence.

Advantages of Sinusoidal Embeddings

Sinusoidal embeddings are used in transformer models to encode positional information in sequences. Here are the key advantages of sinusoidal embeddings:

1. Captures Both Absolute and Relative Positions

- **Absolute Positioning:** Each position in the sequence has a unique encoding derived from sine and cosine functions. This uniqueness ensures that the model can distinguish between different positions, giving it a sense of absolute order within the sequence.
- **Relative Positioning:** Sinusoidal functions allow the model to understand relative positions. By using a combination of sine and cosine with different frequencies, the model can infer the relative distance between two positions. This helps the model capture dependencies in the data.

2. No Need for Learning

- **Fixed and Non-Learnable:** Unlike learned positional embeddings, sinusoidal embeddings are predefined and do not require learning during training. This reduces the number of parameters in the model and simplifies the training process.
- **Efficiency:** Since they are not learned, sinusoidal embeddings do not require additional computations or updates during training, making them computationally efficient.

3. Generalizes to Longer Sequences

- **Scalability:** Sinusoidal embeddings can easily extend to sequences of any length, even those longer than what the model was trained on. Learned embeddings are usually constrained to a fixed maximum length.
- **Smooth Continuity:** The continuous nature of sine and cosine functions means that the positional encodings smoothly generalize beyond the training range. This allows the model to handle sequences of varying lengths without losing positional information.

4. Multi-Scale Representation

- **Multi-Scale Pattern Recognition:** The varying frequencies of the sine and cosine functions (from high to low) enable the model to capture both short-range (local) and long-range (global) dependencies within the sequence.
- **Geometric Progression of Wavelengths:** The geometric progression of wavelengths means that the model can focus on different scales of positional relationships in the sequence, enhancing its ability to understand the data.

5. Mathematical Simplicity and Trigonometric Properties

- **Trigonometric Properties:** Sine and cosine functions have well-known mathematical properties, such as periodicity and the ability to represent complex patterns through linear combinations. These properties make it easier for the model to compute relationships between positions using simple linear transformations.
- **Differentiability:** Sinusoidal functions are smooth and differentiable, which makes them suitable for gradient-based optimization methods used in training deep neural networks.

6. Avoids Overfitting

- **Fixed Nature Reduces Overfitting:** Since sinusoidal embeddings are fixed and not learned, they do not introduce additional learnable parameters into the model. This reduces the risk of overfitting, especially when training on smaller datasets.
- **Stable Positional Information:** The fixed nature of sinusoidal embeddings ensures that positional information remains stable throughout training, helping the model focus on learning meaningful patterns in the input data.

7. Compatibility with Different Input Types

- **Universal Applicability:** While originally designed for textual sequences in natural language processing, sinusoidal embeddings are compatible with other sequential data types, such as audio signals or time series data, where the notion of order and relative positioning is important.