Understanding Transformers

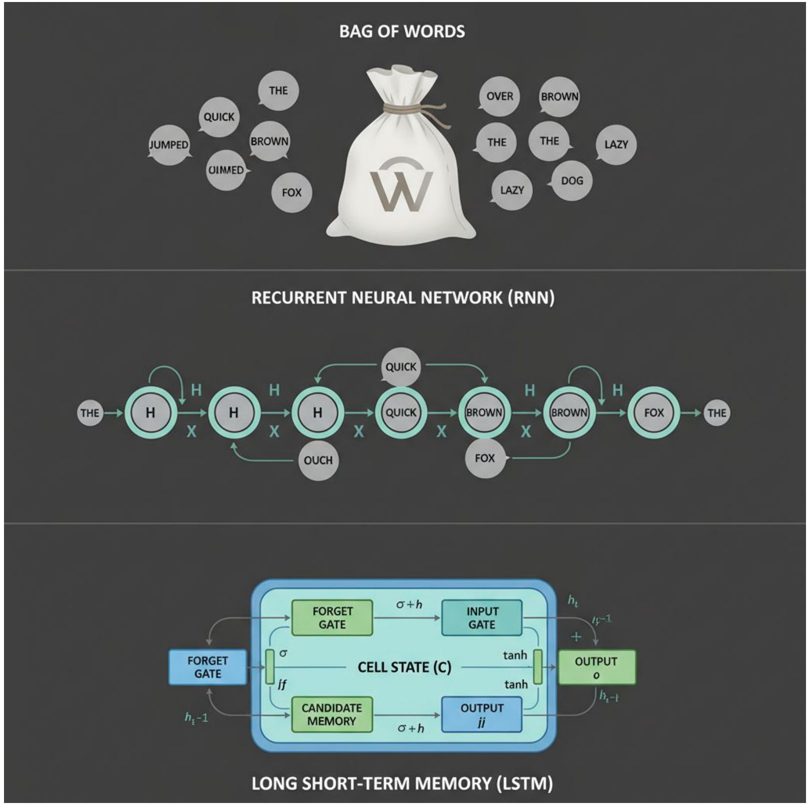
A Step-by-Step Intuition

Before Transformers – The Limits of Earlier Methods

Bag of Words: Treat words independently, miss word order and context

RNN & LSTM: Sequence models that capture context but struggle with long-range dependencies and training complexity

Challenges: Gradient issues, long sequence memory decay, slow training

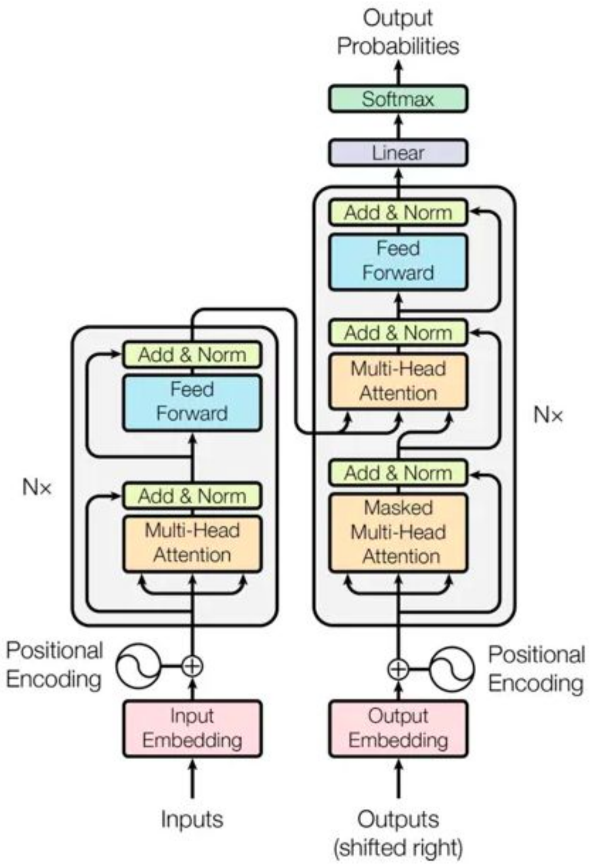


Key Innovation – The Transformer Architecture

Replaces **sequential processing with parallel process**

Introduces **self-attention & positional encoding**

Composed of Encoder and Decoder stacks



Positional Encoding - Keeping Word Order Intact

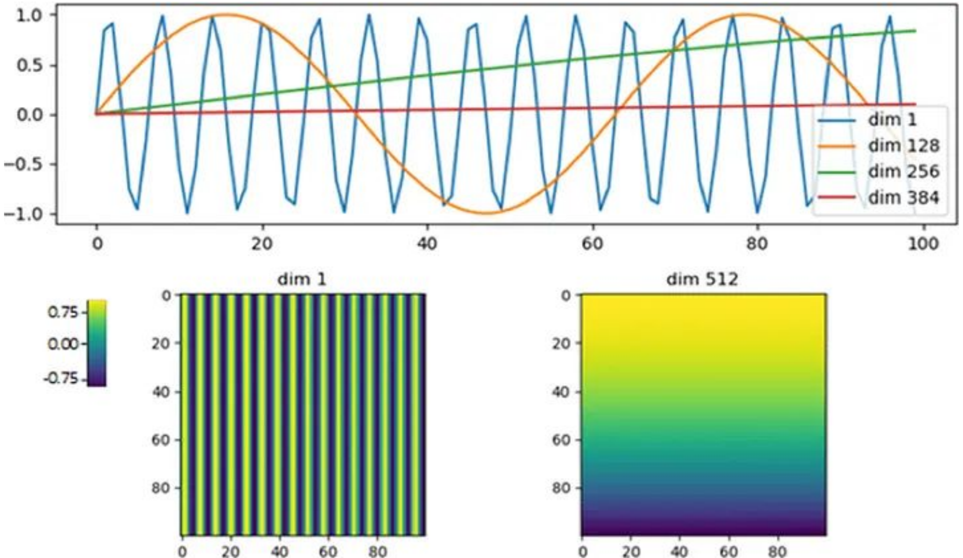
Words get **numerical position** info added to embeddings

Enables **parallel processing** without losing order context

Visualize **sinusoidal encoding** pattern

$$PE_{(i,2dim)} = \sin\left(i/10000^{2dim/d_{\text{model}}}\right)$$

$$PE_{(i,2dim+1)} = \cos\left(i/10000^{2dim/d_{\text{model}}}\right)$$



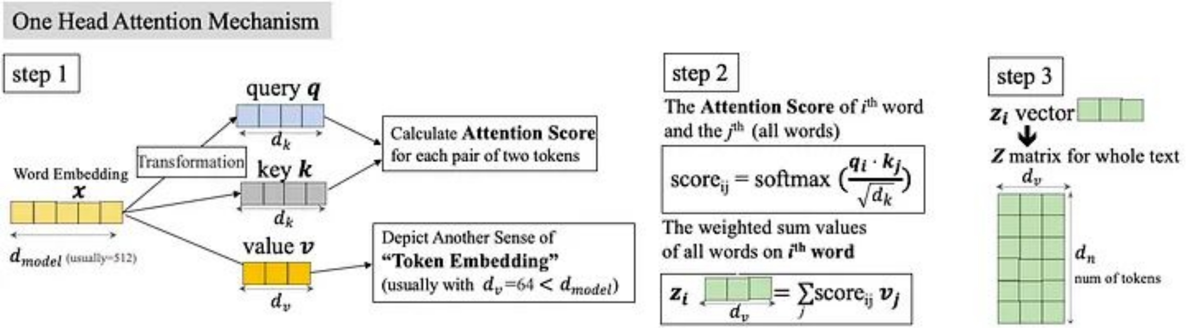
Self-Attention Intuition

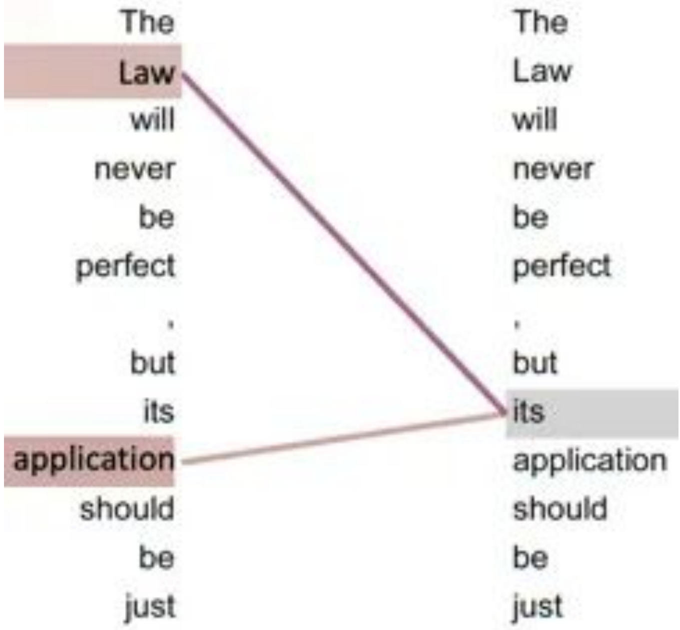
Words look at all other words to decide focus

Use of **Query, Key, and Value** vectors to weight importance

Calculate **attention scores** for each word pair

Weighted sum of values creates context-aware word representation



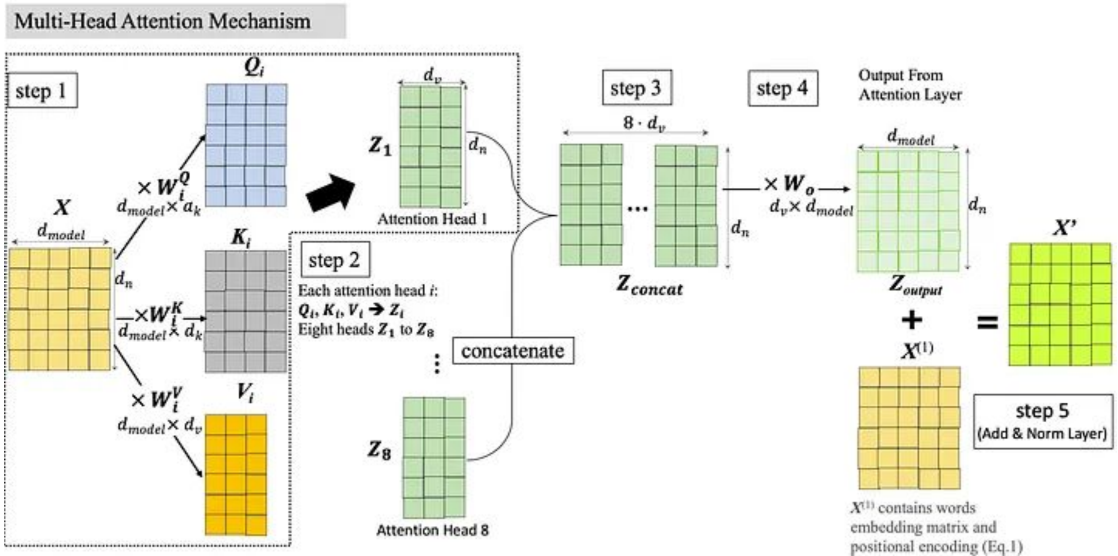


Multi-Head Attention for Deeper Understanding

Multiple attention heads run in parallel

Each head captures different types of relationships

Heads combined to form a comprehensive context-aware representation

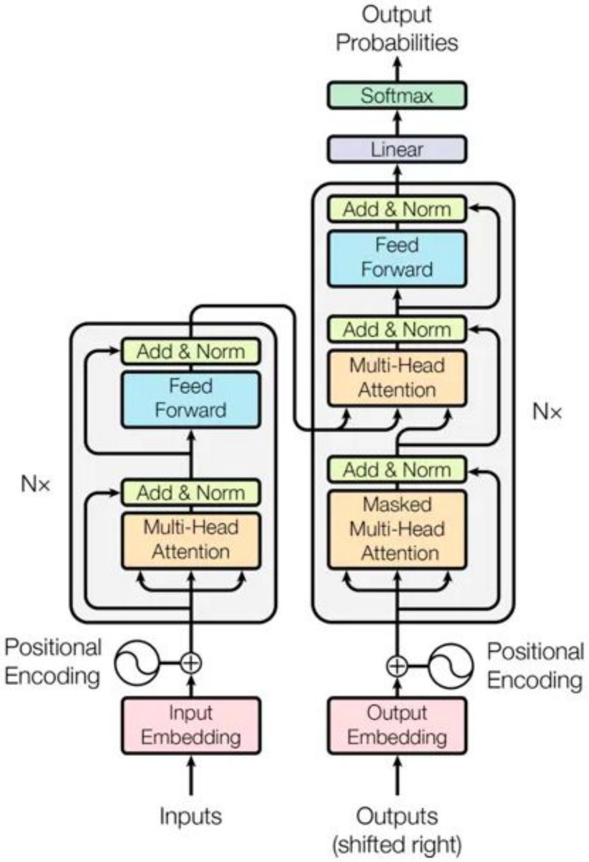


Add & Norm + Feed-Forward Layers

Residual connections add original word info back to attention output

Layer normalization stabilizes learning

Fully connected feed-forward layers for nonlinear transformations

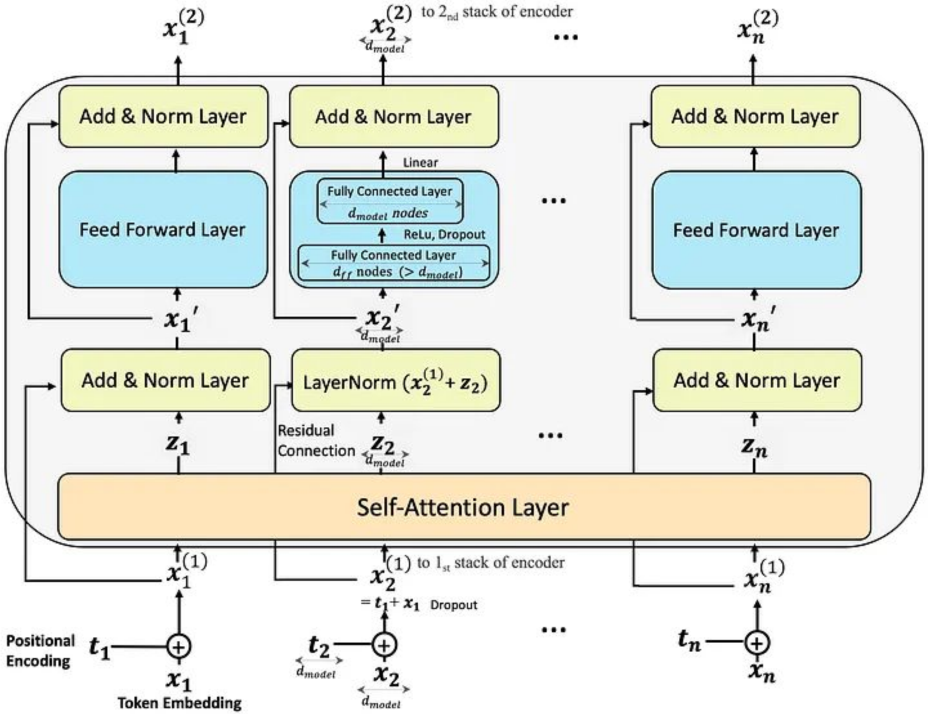


Putting It All Together — Encoder Stacks

Encoder stacks repeat self-attention + add & norm + feed-forward layers N times

Each layer builds richer word-context representations

Final encoder output fed to decoder

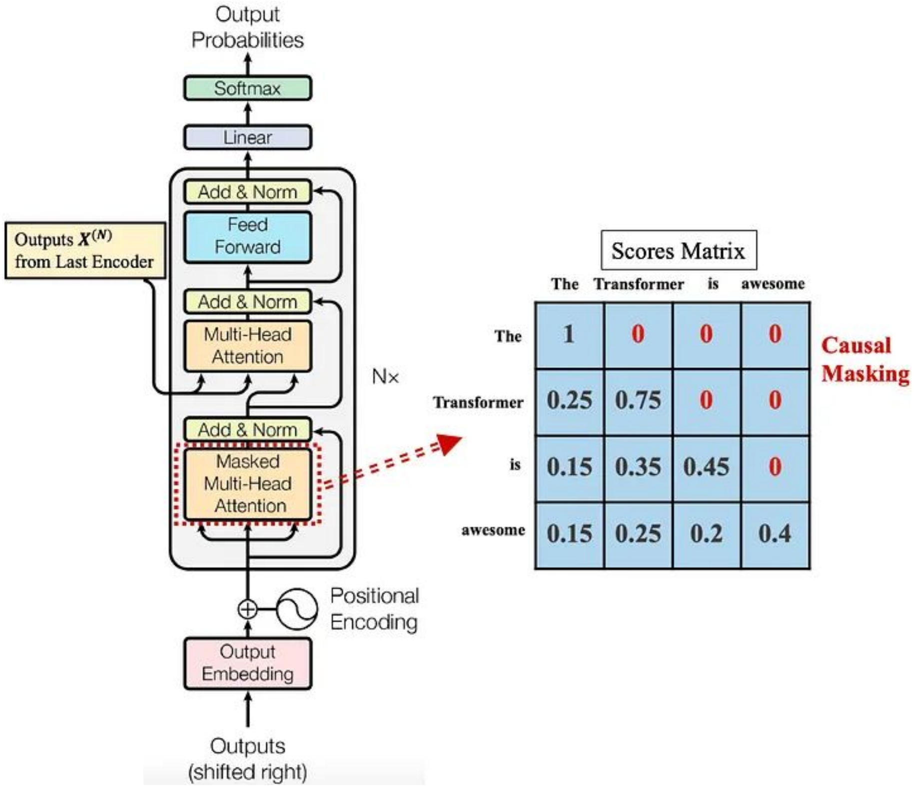


Decoder Overview & Differences

Decoder generates outputs one word at a time

Uses masked self-attention to prohibit future word peeks (causal masking)

Cross-attention references encoder outputs to stay contextually relevant

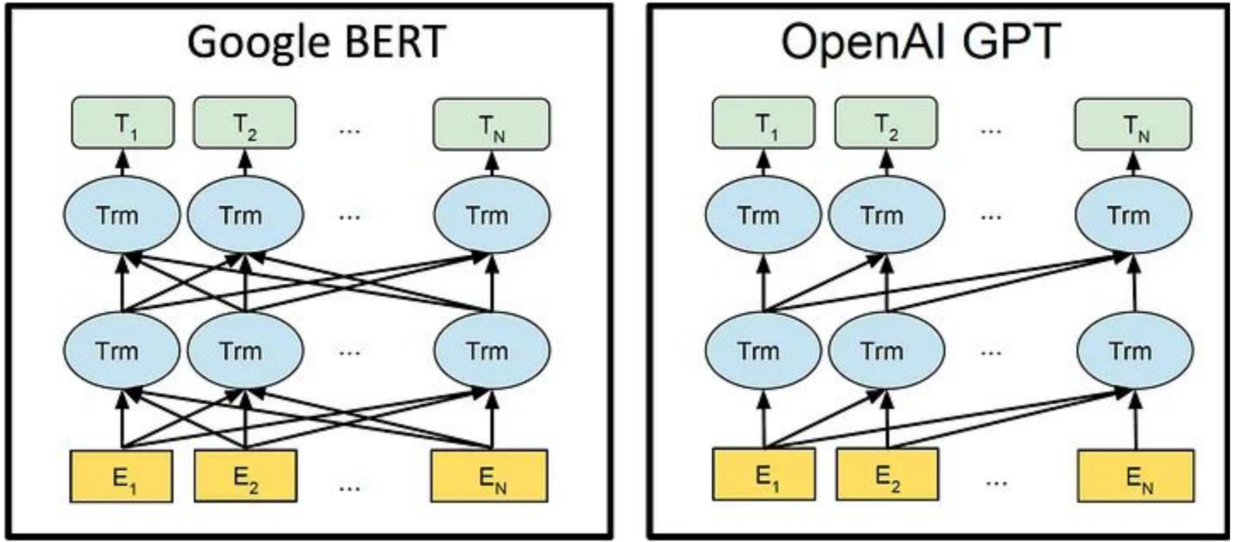


Transformer Models in Practice — BERT & GPT

BERT: Encoder-only, bidirectional for understanding context better

GPT: Decoder-only, left-to-right for generation tasks

Both revolutionized NLP applications



Summary & Intuition Recap

Transformers enable **parallel processing** and **long-range dependency** modeling

Self-attention **dynamically weighs word relationships**

Multi-head attention captures **diverse semantic aspects**

Positional encoding **preserves** word order

Encoder-decoder builds powerful **context-aware** generation capability

Quiz Time!

1. What is the main innovation that enables Transformers to handle long-range dependencies effectively?

a) Using convolutional neural networks

b) Using self-attention mechanisms

c) Using recurrent neural networks

d) Using max pooling

Quiz Time!

2. Why do Transformers add positional encodings to word embeddings?

a) To memorize the words better

b) To enable parallel processing while retaining word order information

c) To reduce the size of the input data

d) To increase model complexity

Quiz Time!

3. What is “multi-head attention” in a Transformer?

a) Using multiple layers of feed-forward networks

b) Using several parallel self-attention mechanisms to capture different aspects of context

c) Dividing the input into multiple segments

d) Applying attention on multiple sentences simultaneously

Quiz Time!

4. In the Transformer decoder, what purpose does masking serve during self-attention?

a) To speed up computation

b) To prevent the model from looking at future words it hasn't generated yet

c) To filter out irrelevant words

d) To normalize the attention scores

Quiz Time!

5. Which Transformer-based model uses only the encoder stack for tasks like understanding text context?

a) GPT

b) BERT

c) T5

d) Transformer XL