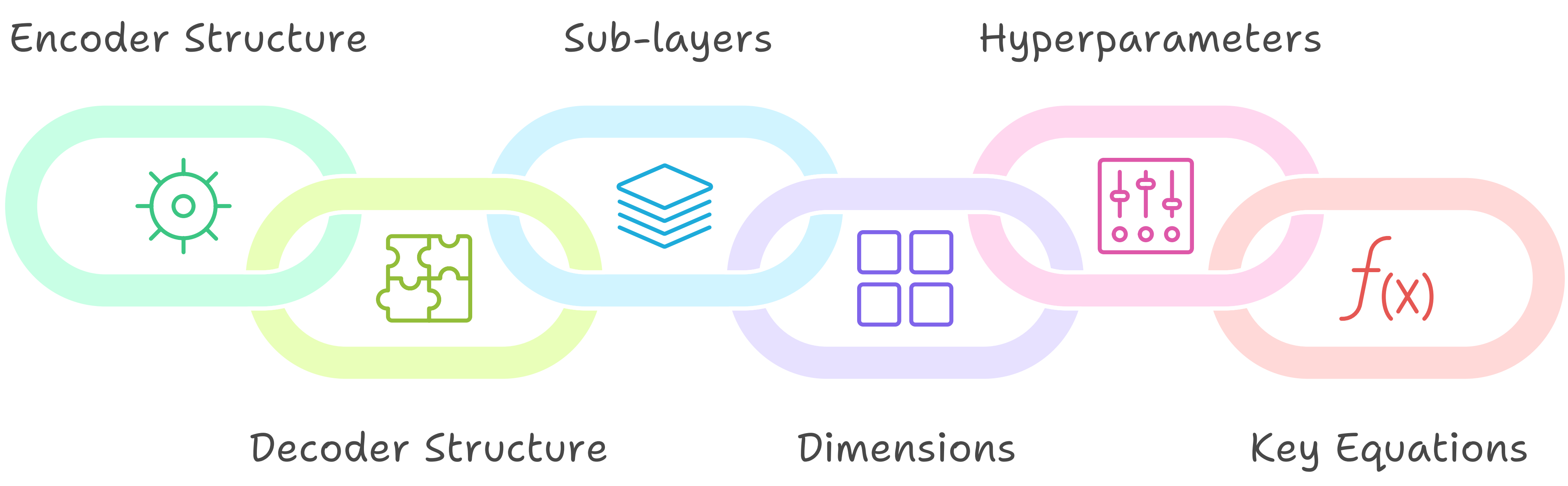
Transformer Architecture



⚙️!! Standard Hyperparameters

☑ Encoder

Model Parameters and Values

Parameter	Value
Model dimension	d_model = 512
Feedforward hidden	d_ff = 2048
Attention heads	8
Dimension per head	d_k = d_q = d_v = 64
Vocabulary size	V
Sequence length	T
Batch size	B

The encoder is repeated N times (commonly 6)

Block Structure:

1. Input Embedding + Positional Encoding

> Input: token IDs (B, T)
> After embedding: (B, T, 512)
> After adding positional encoding: (B, T, 512)

2. Multi-Head Self-Attention

Inputs: (B, T, 512)
Project to Q, K, V: Q = XW_Q,
K = XW_K,
V = XW_V
→ each (B, T, 512)

Split into 8 heads: each head gets (B, T, 64)

Attention scores: QK?/squrt(64) →(B,T,T)
Softmax → weighted sum with V: (B, T, 64)
Concat heads: (B, T, 512)
Linear projection: (B, T, 512)

3. Add & Layer Normalization

Residual connection
Output: (B, T, 512)

4. Position-wise Feedforward Network

Linear 1: 512 → 2048 → (B, T, 2048)
ReLU
Linear 2: 2048 → 512 → (B, T, 512)

5. Add & Layer Normalization

Residual connection
Output: (B, T, 512)

This block is repeated N times, passing (B, T, 512) forward each time.

☑ Decoder

The decoder is also repeated N times (commonly 6).

Block Structure:

1. Input Embedding + Positional Encoding

> Input tokens: (B, T_dec)
> Embedded: (B, T_dec, 512)

1 Masked Multi-Head Self-Attention

Q, K, V from decoder input: (B, T_dec, 512)
Split into 8 heads: (B, T_dec, 64)
Mask future tokens
Scores: QK?/squrt(64) →(B, T_dec, T_dec)
Softmax → weighted sum with V: (B, T, 64)
Concat heads: (B, T_dec, 512)
Linear projection: (B, T_dec, 512)

2 Add & Layer Normalization

Residual connection
Output: (B, T_dec, 512)

3 Cross-Attention (Encoder-Decoder Attention)

Q from decoder (B, T_dec, 512)
K, V from encoder output (B, T, 512)
Project Q/K/V to 8 heads: each (B, T_dec, T)
Scores: QK?/squrt(64) →(B, T_dec, T_dec)
Softmax → weighted sum with V
Concat heads: (B, T_dec, 512)
Linear projection: (B, T_dec, 512)

4 Add & Layer Normalization

Residual connection
Output: (B, T_dec, 512)

5 Position-wise Feedforward

Linear 1: 512 → 2048 → (B, T_dec, 2048)
ReLU
Linear 2: 2048 → 512 → (B, T_dec, 512)

6 Add & Layer Normalization

Residual connection
Output: (B, T_dec, 512)

Final Linear & Softmax

Linear: 512 → V
Output logits: (B, T_dec, V)
Softmax over vocabulary
Final probabilities: (B, T_dec, V)

Position Encoding

Since self-attention does not preserve order, positional encodings are added to embeddings:

PE (pos,2i) = sin(pos/(10000^2i/d_model))
PE (pos,2i+1) = cos(pos/(10000^2i/d_model))

with shape (1, T, 512) broadcast to batch.

Click on the below to understand forward/backward propagation

Forward Propagation

Backward Propagation