Understanding the Transformer Architecture

A **Transformer** is a revolutionary deep learning model designed to understand and generate human-like text with remarkable accuracy. At its core, it consists of **two main components** that work in harmony:

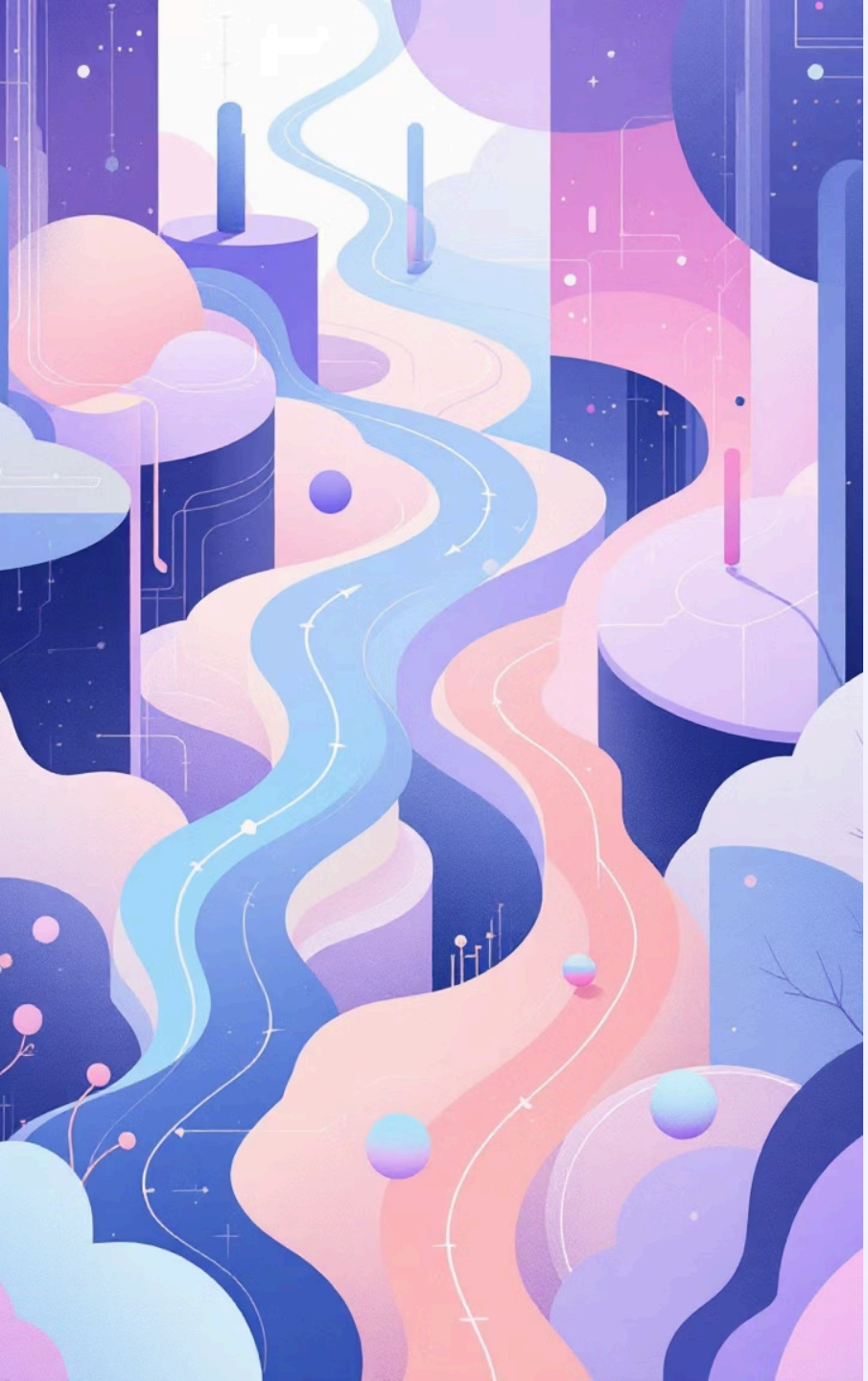
🧩 Encoder

Reads and comprehends the input sentence, extracting meaning and context from the text.

🗣️ Decoder

Leverages that understanding to generate the output sentence, word by word.

Together, they form a powerful pipeline: Input sentence → Encoder → Decoder → Output sentence



The Encoder: Understanding the Input

01

Input Embedding

Each word is transformed into a **vector** (a list of numbers called embeddings) that captures its semantic meaning.

🧠 Example: "Dog" → [0.2, 0.8, 0.4, ...] and "Cat" → [0.3, 0.9, 0.2, ...]

02

Positional Encoding

Since Transformers process all words simultaneously, we add **positional information** to preserve word order.

📘 Formula: **Input Vector = Embedding + Positional Encoding**

03

Encoder Layers

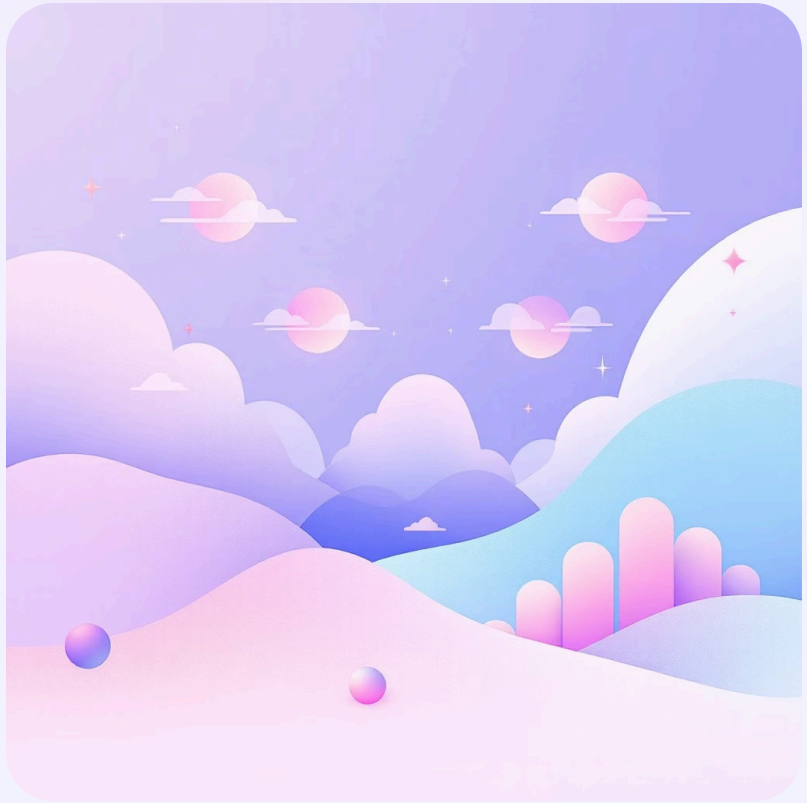
The encoder contains **N stacked layers** (typically 6 or 12), each with Multi-Head Self-Attention and Feed Forward networks.

Multi-Head Self-Attention Mechanism

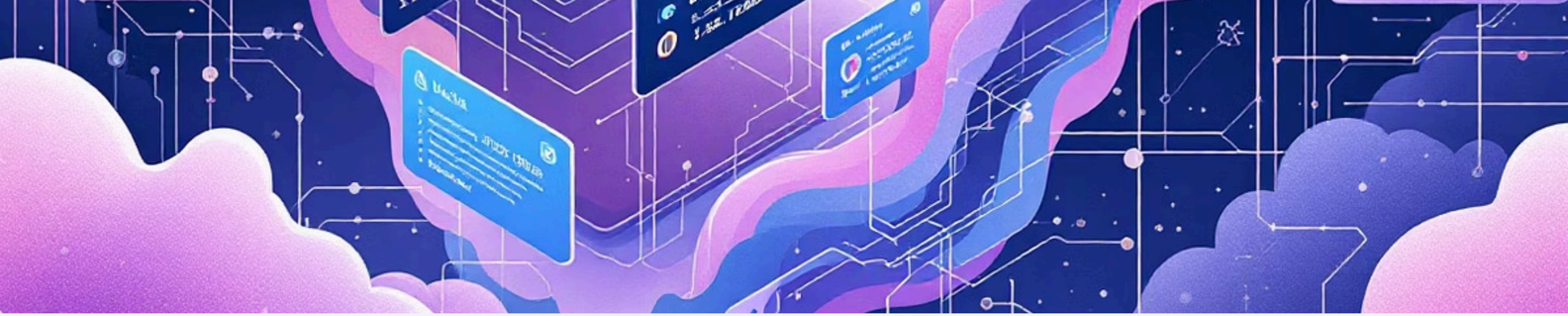
This is where the magic happens. In the **Multi-Head Self-Attention** layer, each word examines *all other words* in the sentence—including itself—to understand relationships and context.

Example: In the sentence "The cat sat on the mat," the word "cat" learns to pay attention to "sat" and "mat" to understand the complete action.

✅ This mechanism allows the model to capture complex dependencies and contextual nuances that traditional models miss.



Multiple Heads: The "multi-head" aspect means attention is computed several times in parallel, each focusing on different aspects of word relationships.



Feed Forward & Normalization

Feed Forward Network

After attention, each word's vector passes through a small **neural network** independently.

- Identifies deeper patterns in the data
- Applies the same transformation to each position
- Adds non-linear complexity to the model

Add & Norm

Each sub-layer in the encoder includes two critical operations:

- **Add:** Residual connection that adds the original input back
- **Norm:** Layer normalization that stabilizes training

This design helps the model learn faster and prevents gradient issues.

The Decoder

Generating the Output Sequence

The **decoder** takes the encoder's rich understanding of the input and transforms it into meaningful output text—one word at a time.

While structurally similar to the encoder, the decoder has additional mechanisms that allow it to:

- Generate output sequentially
- Attend to encoder representations
- Prevent looking ahead during training

Decoder Input & Masked Attention

Output Embedding + Positional Encoding

The decoder begins with previously generated words (shifted right by one position), which are converted into embeddings and enhanced with positional encodings.

Masked Multi-Head Attention

This attention layer can only look at **previous words**, not future ones. The model predicts one word at a time without "peeking ahead."

🛑 This masking prevents the model from cheating during training.

Example: When predicting the 3rd word, the decoder only has access to information from the 1st and 2nd words.

Cross-Attention & Processing

Encoder-Decoder Attention

Here's where the decoder **connects with the encoder**. The decoder attends to the encoder's output to determine which parts of the input sentence are most relevant for generating the next word.

Example: When translating "I love apples" → "J'aime les pommes," the decoder focuses on "love" when generating "aime."

Feed Forward Network

Identical to the encoder's structure, this fully connected network combines all the contextual information gathered from both masked attention and encoder-decoder attention.

It helps synthesize complex patterns for accurate predictions.

Final Prediction Layer

Linear Layer

Transforms the decoder's output vectors into scores for every word in the vocabulary.

Softmax Layer

Converts scores into probabilities—each representing the likelihood of being the next word.

Word Selection

The word with the **highest probability** becomes the next output word.

This process repeats iteratively until a special end-of-sequence token is generated, signaling completion.

How Encoder & Decoder Work Together

Step	Encoder	Decoder
1️⃣ Input	Reads and processes input sentence	Takes previously generated words
2️⃣ Understanding	Extracts meaning through self-attention	Predicts next word using masked attention
3️⃣ Output	Sends contextualized representation to decoder	Uses encoder information to generate output sentence

🎯 **The complete process:** The encoder creates a deep understanding of the input, while the decoder leverages this understanding to generate the output sequence word by word. This process repeats until the entire sentence is produced.

> "The Transformer architecture has revolutionized natural language processing, enabling models to understand context and generate human-like text with unprecedented accuracy."