Step 2 — Masked Multi-Head Self-Attention

How the decoder learns to "pay attention" only to previous words

Quick Recap — What Happens in the Decoder

The decoder generates the output sequence *one word at a time*. For instance, when translating "The cat sat" to French "Le chat s'est assis", the model must predict each subsequent word based on:

- Previously generated words in the sequence
- Rich contextual information from the encoder

To ensure proper learning and prevent the model from "seeing the future" during training, we apply a crucial technique called **masking**.

Why Masking Is Needed

Understanding the sequential nature of decoding reveals why masking is essential:

01

First Step

Input: <START> → Model predicts: "Le"

02

Second Step

Input: <START> Le → Model predicts: "chat"

03

Third Step

Input: <START> Le chat → Model predicts: "s'est"

💡 **Critical Problem:** If the model could *see* the word "chat" whilst trying to predict it, that would constitute cheating. Therefore, we hide (mask) future words during training, ensuring the model learns exclusively from past context—mirroring how we naturally read from left to right.

What Is a Mask?

A **mask** is a matrix that instructs the model which words to attend to and which to ignore. This structure is known as a **look-ahead mask** or **causal mask**.

For the sequence ["Le", "chat", "est"], the mask determines visibility:

- ✅ = word is visible (allowed to attend)
- ❌ = word is hidden (masked)

Current Word →	Le	chat	est
Le	✅	❌	❌
chat	✅	✅	❌
est	✅	✅	✅

How Masking Works Mathematically

Each word undergoes linear transformations to generate three crucial vectors:

Q (Query)

What the word is looking for

K (Key)

What the word contains

V (Value)

The actual meaning

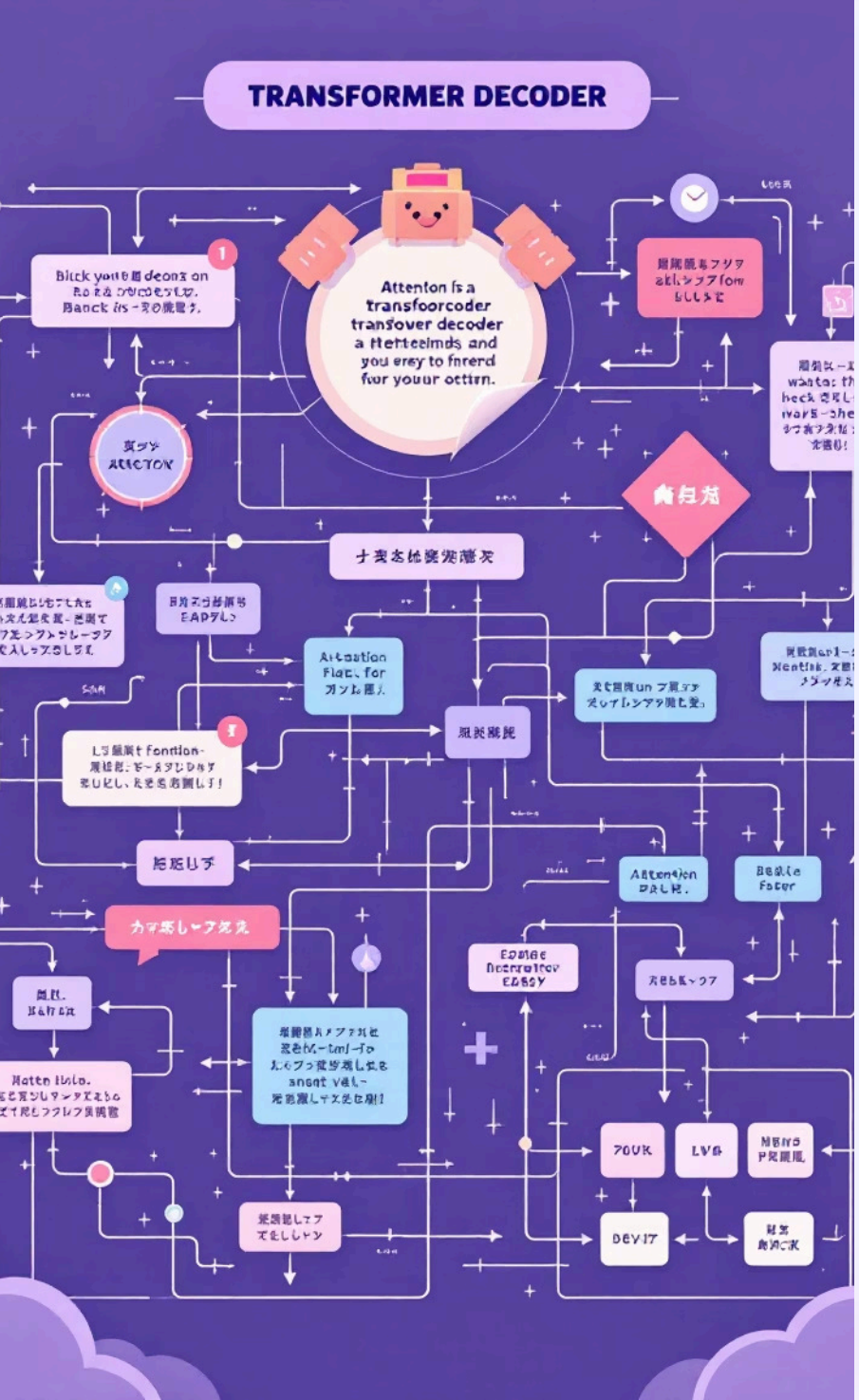
We compute **attention scores** using the following formula:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + \text{Mask}\right)V$$

Where:

- **Q, K, V:** Query, Key, Value matrices
- **dk:** Dimension of key vectors (used for scaling)
- **Mask:** Matrix with 0 for allowed positions and −∞ for future positions

🧠 **Effect:** When we add the mask, future positions become −∞. After applying softmax, those probabilities become **exactly 0**, causing the model to completely ignore them.



Step-by-Step Example

Let's examine how masking works with the sentence: **"Le chat est"**

1

Step 1: Word "Le"

Can only attend to itself—no previous context available yet

Step 2: Word "chat"

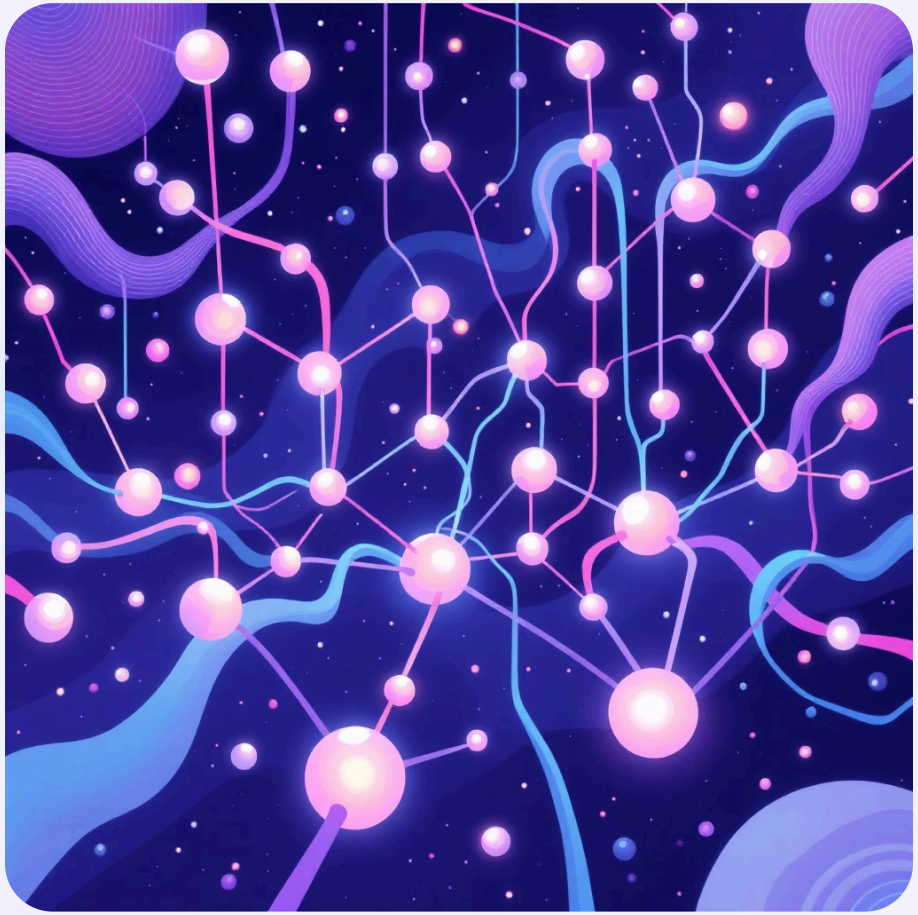
Can see both "Le" and "chat"—building contextual understanding

Step 3: Word "est"

Can see all previous words—full context from the sequence so far

This mechanism ensures that **each word attends only to earlier words**, never to future ones, maintaining the causal nature of language generation.

What Happens After Masking



Once attention weights are calculated, the model proceeds through these stages:

1. Each word's value vectors (V) are multiplied by their corresponding attention weights
2. The result is a new vector representing the word in its **context accumulated so far**
3. Contextual enrichment example:
    - "chat" now carries information from "Le"
    - "est" carries information from both "Le" and "chat"

The model progressively builds a richer, more nuanced context as it advances through the sequence, enabling sophisticated language understanding.

Multi-Head Attention — The "Multi" Part

Rather than relying on a single attention mechanism, the decoder employs **multiple heads** (typically 8) operating in parallel.

1

Grammar Patterns

Head 1 focuses on syntactic structures

2

Subject–Object Relations

Head 2 tracks entity relationships

3

Verb–Tense Agreement

Head 3 ensures temporal consistency

4

Word Order

Head 4 maintains sequential logic

5

Other Contextual Cues

Heads 5–8 capture additional linguistic nuances

All heads' outputs are then **concatenated** and passed through a linear layer, combining diverse perspectives into a unified representation.

Why Masked Multi-Head Attention Is Crucial

Prevents Cheating

Eliminates look-ahead bias during training

Enforces Causality

Forces model to learn left-to-right dependencies

Natural Prediction

Helps model generate text like humans—word by word

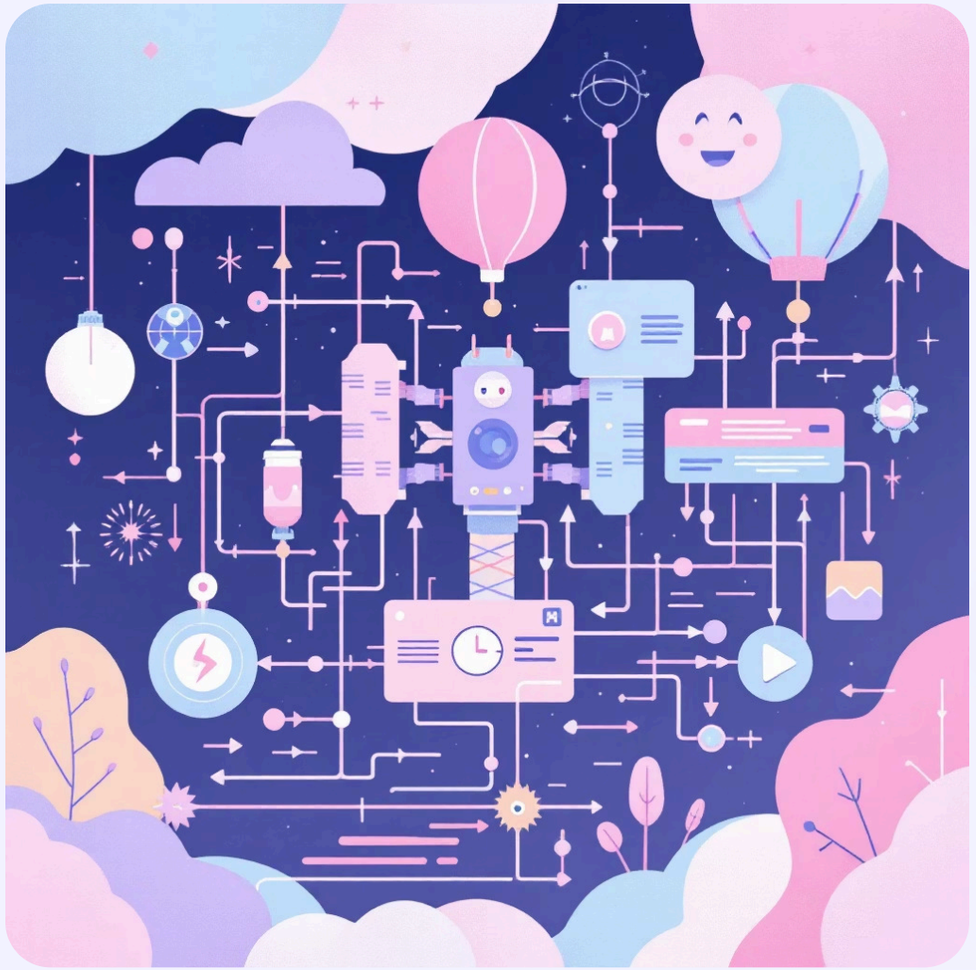
Enables Real Inference

Functions properly when future words are genuinely unknown

Critical Insight: Without masking, the model would demonstrate unrealistically high performance during training but fail catastrophically during real-world usage when actual future tokens are unavailable.

Summary

Concept	Meaning
Mask	Blocks attention to future words
Look-ahead Mask	Ensures word *i* sees only positions ⩽ *i*
Multi-head	Multiple attention mechanisms in parallel
Goal	Learn to predict next word without cheating



In short: Masked Multi-Head Attention helps the decoder focus exclusively on what it already knows—precisely like a person writing a sentence word by word, ensuring authentic, causal language generation.