**🔎 Why CNNs were once popular for embeddings**

Before Transformers became dominant, people often used **CNNs (convolutional neural nets)** or **RNNs (LSTMs, GRUs)** to turn sequences (text, audio) into compact **vector encodings**:

* **CNNs:** good at detecting local patterns (e.g., character n-grams, local word windows).
* **RNNs:** good at capturing sequential dependencies (order).

So if you were building a sentence encoder in, say, 2015–2017, you’d almost certainly use CNNs or RNNs.

**🔎 What Transformers do instead**

In Transformers (both encoder and decoder):

1. **Embeddings layer**  
   Each token (word, subword, or character) is mapped to a dense vector from a learned lookup table (an **embedding matrix**).  
   Example: "dog" → [0.23, -0.77, …, 0.44] (size 768 in BERT-base).  
   👉 This step is **not CNN or RNN** — it’s just a parameter lookup.
2. **Positional encoding**  
   Since embeddings themselves carry no order information, Transformers add a **positional encoding** (sinusoidal or learned) to each embedding vector.  
   👉 This replaces the sequential inductive bias that RNNs or CNNs naturally provide.
3. **Self-attention layers**  
   Instead of convolutions, Transformers rely on **multi-head self-attention** to model relationships between tokens across the whole sequence.
   * Attention can connect distant words directly (e.g., subject ↔ verb, even far apart).
   * CNNs are local by default; attention is global.
4. **Feed-forward layers**  
   Each token is further refined by a position-wise feed-forward MLP inside each Transformer block.

**🔎 So why no CNNs inside the Transformer?**

Because:

* The **embedding matrix** already provides token vectors.
* **Positional encoding** injects sequence order.
* **Self-attention** provides global context (more powerful and flexible than CNNs’ local filters).

CNNs *can* be used in hybrid models (e.g., Conformers in speech recognition = CNN + Transformer), but in standard NLP Transformers (BERT, GPT, T5), CNNs are not necessary.

✅ **Key takeaway:**  
Transformers don’t need CNNs for embedding because they:

* learn embeddings directly from a lookup table,
* add position info explicitly, and
* use self-attention (not convolutions) to capture relationships.