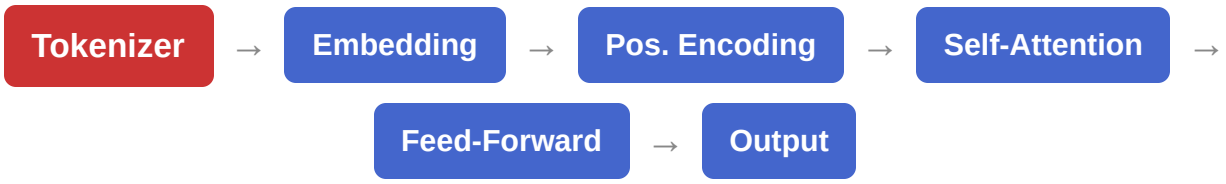
Transformer aus Papier — Schritt 1: Der Tokenizer



Was macht der Tokenizer?
Ein Transformer kann nicht mit Text arbeiten — nur mit Zahlen. Der Tokenizer zerlegt Text in kleine Einheiten (Tokens) und gibt jeder Einheit eine Nummer (ID).

Unser Beispielsatz: „Die Katze sitzt auf der Matte"

Auf den folgenden Seiten findest du **3 Blätter zum Ausdrucken und Ausschneiden**:

Blatt	Methode	Einheit	Anzahl Tokens
Blatt 1	Kein Tokenizer	Ganzer Satz	1
Blatt 2	Word-Level	Wörter	6
Blatt 3	Character-Level	Buchstaben	26

Anleitung:
1. Drucke alle Seiten aus
2. Schneide entlang der gestrichelten Linien (- - -)
3. Schreibe die **ID-Nummer** auf die Rückseite jeder Karte
4. Dreh alle Karten um → das ist dein **Input-Vektor** für den Transformer!

Blatt 1 — Ganzer Satz (kein Tokenizer)

> Nicht schneiden! Der ganze Satz ist eine einzige Einheit.
> **Problem:** Der Transformer kann nichts über einzelne Wörter lernen. Jeder neue Satz bräuchte eine eigene ID — unmöglich bei Millionen von Sätzen.

Die Katze sitzt auf der Matte

→ 1 Token · ID: 0 · Vokabular müsste unendlich groß sein ❌

Blatt 2 — Word-Level Tokenizer

✂️ **Entlang der gestrichelten Linien ausschneiden!**

> Jedes Wort wird ein eigenes Token. Schreib die ID auf die Rückseite.

Die	Katze	sitzt	auf	der
ID: 0	ID: 1	ID: 2	ID: 3	ID: 4

Matte
ID: 5

⬇

Input-Vektor (Rückseite):

0	1	2	3	4

5

> **Vokabular:** {Die, Katze, sitzt, auf, der, Matte} = 6 Einträge
> **Vorteil:** Wenige Tokens pro Satz
> **Nachteil:** „Katzen" (Plural) wäre ein komplett neues Token! Jedes unbekannte Wort = Problem. Das Vokabular müsste riesig sein.

Blatt 3 — Character-Level Tokenizer

✂️ **Jeden Buchstaben einzeln ausschneiden!**

> Jeder Buchstabe wird ein eigenes Token. Gleiche Buchstaben bekommen die gleiche ID. Leerzeichen werden als ␣ dargestellt.

Vokabular (alle verschiedenen Zeichen):

Zeichen	D	i	e	␣	K	a	t	z	s	u	f	d	r	M
ID	0	1	2	3	4	5	6	7	8	9	10	11	12	13

— „Die" —

D	**i**	**e**
ID: 0	ID: 1	ID: 2

— „Katze" —

K	**a**	**t**	**z**	**e**
ID: 4	ID: 5	ID: 6	ID: 7	ID: 2

— „sitzt" —

s	**i**	**t**	**z**	**t**
ID: 8	ID: 1	ID: 6	ID: 7	ID: 6

— „auf" —

a	u	f
ID: 5	ID: 9	ID: 10

— „der" —

d	e	r
ID: 11	ID: 2	ID: 12

— „Matte" —

M	a	t	t	e
ID: 13	ID: 5	ID: 6	ID: 6	ID: 2

Input-Vektor (alle IDs hintereinander):

```
[0, 1, 2, 3, 4, 5, 6, 7, 2, 3, 8, 1, 6, 7, 6, 3, 5, 9, 10, 3, 11, 2, 12,
3, 13, 5, 6, 6, 2]
```

Vokabular: Nur 14 verschiedene Zeichen — winzig!
Vorteil: Jedes Wort darstellbar, auch unbekannte
Nachteil: 29 Tokens für 6 Wörter — der Transformer muss viel mehr verarbeiten und hat es schwerer, Wort-Bedeutungen zu lernen.

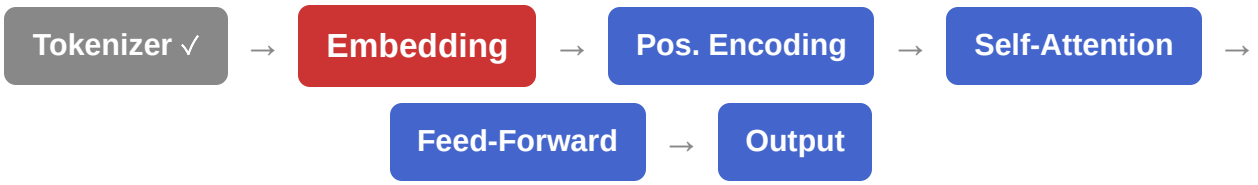
Zusammenfassung & nächster Schritt

> **Was du jetzt vor dir hast:**
> Kärtchen mit Text auf der Vorderseite und Zahlen auf der Rückseite. Dreh alle Karten um → du siehst nur noch Zahlen. **Das ist alles, was der Transformer sieht!**

	Word-Level	Character-Level	Subword (BPE) *
Tokens	6	29 (mit Leerzeichen)	~8-12
Vokabular	riesig (jedes Wort)	winzig (~100 Zeichen)	mittel (~30.000-50.000)
Unbekannte Wörter?	❌ Problem	✅ Kein Problem	✅ Kein Problem
Bedeutung pro Token	✅ Hoch	❌ Niedrig	✅ Gut

** Subword (BPE) ist der Mittelweg, den GPT & Co. nutzen — kommt in Schritt 1b.*

Nächster Schritt: Embedding



Jede Token-ID bekommt jetzt einen **Vektor** — eine Liste von Zahlen, die die „Bedeutung" des Tokens darstellt. Dafür bauen wir eine **Embedding-Tabelle** aus Papier: eine Tabelle, in der du für jede ID eine Zeile mit Zahlen nachschlagen kannst.