Transformer — Komplette Mathematik

Alle Formeln und Berechnungen für „Die Katze sitzt auf der Matte" · 4 Dimensionen · 2 Heads · 1 Block

0 Modell-Parameter

Parameter	Unser Modell	GPT-2 Small	GPT-4 (geschätzt)
Embedding-Dimension (d_{model})	**4**	768	~12.288
Attention Heads (h)	**2**	12	~96
Dimension pro Head ($d_k = d_{model}/h$)	**2**	64	~128
FFN innere Dimension (d_{ff})	**8**	3072	~49.152
Transformer-Blöcke (L)	**1**	12	~120
Vokabular (V)	**6**	50.257	~100.000
Kontext-Länge (n)	**6**	1.024	128.000+

1 Tokenizer (Word-Level)

```
Text → Split(Leerzeichen) → Token-IDs aus Vokabular
```

Wort	Die	Katze	sitzt	auf	der	Matte
Token-ID	0	1	2	3	4	5

Input-Sequenz X = [0, 1, 2, 3, 4, 5] (Länge n = 6)

2 Embedding-Lookup

$$E_i = \text{EmbeddingTable}[\text{token\_id}_i]$$

Tabelle hat Größe V × d_{model} = 6 × 4

Token	ID	d_1	d_2	d_3	d_4
Die	0	0.9	0.1	0.0	0.1
Katze	1	0.0	0.9	0.1	0.2
sitzt	2	0.0	0.1	0.9	0.0
auf	3	0.5	0.0	0.3	0.4
der	4	0.9	0.1	0.0	0.1
Matte	5	0.0	0.0	0.0	0.9

3 Positional Encoding

$$PE(pos, 2i) = \sin(pos / 10000^{2i/d}) \quad PE(pos, 2i+1) = \cos(pos / 10000^{2i/d})$$

d = 4 → Teiler: i=0 → $10000^{0} = 1$, i=1 → $10000^{0.5} = 100$

Pos	Wort	sin(pos/1)	cos(pos/1)	sin(pos/100)	cos(pos/100)
0	Die	0.000	1.000	0.000	1.000
1	Katze	0.841	0.540	0.010	1.000
2	sitzt	0.909	-0.416	0.020	1.000
3	auf	0.141	-0.990	0.030	1.000
4	der	-0.757	-0.654	0.040	0.999
5	Matte	-0.959	0.284	0.050	0.999

$$X_{final} = Embedding + PE \quad \text{(elementweise Addition)}$$

Ergebnis: Input-Matrix X (6 × 4) — gerundet

Pos	Token	d_1	d_2	d_3	d_4
0	**Die**	0.9	1.1	0.0	1.1
1	**Katze**	0.8	1.4	0.1	1.2
2	**sitzt**	0.9	-0.3	0.9	1.0
3	**auf**	0.6	-1.0	0.3	1.4
4	**der**	0.1	-0.6	0.0	1.1
5	**Matte**	-1.0	0.3	0.1	1.9

4 Masked Multi-Head Self-Attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left((Q \cdot K^T) / \sqrt{d_k} + \text{Mask} \right) \cdot V$$

$d_k = 2, \quad \sqrt{d_k} = \sqrt{2} \approx 1.41$

W-Matrizen (4 × 2 pro Head)

Head 1

	W_{Q1}		W_{K1}		W_{V1}	
d_1	1	0	0	0	0	0
d_2	0	1	0	0	1	0
d_3	0	0	1	0	0	0
d_4	0	0	0	1	0	1

$Q_1=[d_1,d_2]$ $K_1=[d_3,d_4]$ $V_1=[d_2,d_4]$

Head 2

	W_{Q2}		W_{K2}		W_{V2}	
d_1	0	0	1	0	1	0
d_2	0	0	0	1	0	0
d_3	1	0	0	0	0	1
d_4	0	1	0	0	0	0

$Q_2=[d_3,d_4]$ $K_2=[d_1,d_2]$ $V_2=[d_1,d_3]$

Q, K, V für alle Tokens

Token	Q_1		K_1		V_1		Q_2		K_2		V_2	
Die	0.9	1.1	0.0	1.1	1.1	1.1	0.0	1.1	0.9	1.1	0.9	0.0
Katze	0.8	1.4	0.1	1.2	1.4	1.2	0.1	1.2	0.8	1.4	0.8	0.1
sitzt	0.9	-0.3	0.9	1.0	-0.3	1.0	0.9	1.0	0.9	-0.3	0.9	0.9
auf	0.6	-1.0	0.3	1.4	-1.0	1.4	0.3	1.4	0.6	-1.0	0.6	0.3
der	0.1	-0.6	0.0	1.1	-0.6	1.1	0.0	1.1	0.1	-0.6	0.1	0.0
Matte	-1.0	0.3	0.1	1.9	0.3	1.9	0.1	1.9	-1.0	0.3	-1.0	0.1

Causal Mask (Dreiecksmatrix)

$$\text{Mask}(i,j) = 0 \text{ wenn } j \le i, \quad -\infty \text{ wenn } j > i$$

	Die	Kat	sit	auf	der	Mat
Die	0	-∞	-∞	-∞	-∞	-∞
Kat	0	0	-∞	-∞	-∞	-∞
sit	0	0	0	-∞	-∞	-∞
auf	0	0	0	0	-∞	-∞
der	0	0	0	0	0	-∞
Mat	0	0	0	0	0	0

Head 1: Raw Scores = $Q_1 \cdot K_1^T$

> Score(i,j) = $Q_{1i} \cdot K_{1j} = Q_{1i}[0]\times K_{1j}[0] + Q_{1i}[1]\times K_{1j}[1]$

$Q_1 \cdot K_1^T$	Die [0.0,1.1]	Katze [0.1,1.2]	sitzt [0.9,1.0]	auf [0.3,1.4]	der [0.0,1.1]	Matte [0.1,1.9]
Die [0.9,1.1]	1.21	-∞	-∞	-∞	-∞	-∞
Katze [0.8,1.4]	1.54	1.76	-∞	-∞	-∞	-∞
sitzt [0.9,-0.3]	-0.33	-0.27	0.51	-∞	-∞	-∞
auf [0.6,-1.0]	-1.10	-1.14	-0.46	-1.22	-∞	-∞
der [0.1,-0.6]	-0.66	-0.71	-0.51	-0.81	-0.66	-∞
Matte [-1.0,0.3]	0.33	0.26	-0.60	0.12	0.33	0.47

Head 1: Scaled Scores (÷ √2 = 1.41)

÷ 1.41	Die	Katze	sitzt	auf	der	Matte
Die	0.86	-∞	-∞	-∞	-∞	-∞
Katze	1.09	1.25	-∞	-∞	-∞	-∞
sitzt	-0.23	-0.19	0.36	-∞	-∞	-∞
auf	-0.78	-0.81	-0.33	-0.87	-∞	-∞
der	-0.47	-0.50	-0.36	-0.57	-0.47	-∞
Matte	0.23	0.18	-0.43	0.09	0.23	0.33

Head 1: Attention Weights (Softmax pro Zeile)

$$\text{softmax}(z_i) = e^{z_i} / \Sigma_j\, e^{z_j} \quad \text{(nur über sichtbare Positionen, } e^{-\infty} = 0\text{)}$$

Weights_1	Die	Katze	sitzt	auf	der	Matte
Die	**1.000**	0	0	0	0	0
Katze	0.460	**0.540**	0	0	0	0
sitzt	0.276	0.288	**0.436**	0	0	0
auf	0.274	0.266	0.306	**0.154**	0	0
der	0.211	0.205	0.215	0.164	**0.205**	0
Matte	0.190	0.181	0.123	0.165	0.190	**0.151**

Head 1: Output = $\text{Weights}_1 \cdot V_1$

$$\text{Output}_{1i} = \Sigma_j\, \text{weight}_1(i,j) \times V_{1j}$$

Token	Output_1 [v_1, v_2]	Berechnung (Hauptbeiträge)
Die	**[1.10, 1.10]**	1.0×[1.1,1.1]
Katze	**[1.26, 1.15]**	0.46×[1.1,1.1] + 0.54×[1.4,1.2]
sitzt	**[0.58, 1.09]**	0.28×[1.1,1.1] + 0.29×[1.4,1.2] + 0.44×[-0.3,1.0]
auf	**[0.38, 1.14]**	0.27×[1.1,1.1] + 0.27×[1.4,1.2] + 0.31×[-0.3,1.0] + 0.15×[-1.0,1.4]
der	**[0.26, 1.12]**	Alle 5 sichtbaren, relativ gleichverteilt
Matte	**[0.27, 1.27]**	Alle 6, relativ gleichverteilt

Head 2: Raw Scores = $Q_2 \cdot K_2^T$ (nach Mask + ÷ √2)

$Scores_2$ (skaliert)	Die	Katze	sitzt	auf	der	Matte
Die Q_2=[0.0,1.1]	0.78	-∞	-∞	-∞	-∞	-∞
Katze Q_2=[0.1,1.2]	1.00	1.25	-∞	-∞	-∞	-∞
sitzt Q_2=[0.9,1.0]	1.35	1.50	0.36	-∞	-∞	-∞
auf Q_2=[0.3,1.4]	1.28	1.57	-0.11	0.87	-∞	-∞
der Q_2=[0.0,1.1]	0.78	1.09	-0.23	0.42	-0.40	-∞
Matte Q_2=[0.1,1.9]	1.54	1.94	0.22	1.28	0.80	0.48

Head 2: Attention Weights (Softmax)

$Weights_2$	Die	Katze	sitzt	auf	der	Matte
Die	**1.000**	0	0	0	0	0
Katze	0.438	**0.562**	0	0	0	0
sitzt	0.359	**0.418**	0.223	0	0	0
auf	0.273	**0.366**	0.068	0.293	0	0
der	0.252	**0.344**	0.092	0.176	0.136	0
Matte	0.231	**0.343**	0.062	0.178	0.110	0.076

Head 2: Output = $Weights_2 \cdot V_2$

Token	$Output_2$ [v_1, v_2]
Die	**[0.90, 0.00]**
Katze	**[0.84, 0.06]**
sitzt	**[0.86, 0.24]**
auf	**[0.78, 0.17]**
der	**[0.73, 0.13]**
Matte	**[0.56, 0.13]**

Concatenation: Head 1 ⊕ Head 2 = Attention-Output (4D)

$$MultiHead(X) = Concat(Head_1, Head_2) \cdot W_0$$

$W_O = I_4$ (Identität) in unserem Modell

Token	Head 1	Head 2	Attention-Output [d_1, d_2, d_3, d_4]
Die	[1.10, 1.10]	[0.90, 0.00]	**[1.10, 1.10, 0.90, 0.00]**
Katze	[1.26, 1.15]	[0.84, 0.06]	**[1.26, 1.15, 0.84, 0.06]**
sitzt	[0.58, 1.09]	[0.86, 0.24]	**[0.58, 1.09, 0.86, 0.24]**
auf	[0.38, 1.14]	[0.78, 0.17]	**[0.38, 1.14, 0.78, 0.17]**
der	[0.26, 1.12]	[0.73, 0.13]	**[0.26, 1.12, 0.73, 0.13]**
Matte	[0.27, 1.27]	[0.56, 0.13]	**[0.27, 1.27, 0.56, 0.13]**

5 Add & Layer Norm (1. Mal)

$$Z = \text{LayerNorm}(X + \text{MultiHead}(X))$$

Add: X + Attention-Output

Token	X (Input)	+ Attn-Output	= Summe
Die	[0.9, 1.1, 0.0, 1.1]	[1.10, 1.10, 0.90, 0.00]	**[2.00, 2.20, 0.90, 1.10]**
Katze	[0.8, 1.4, 0.1, 1.2]	[1.26, 1.15, 0.84, 0.06]	**[2.06, 2.55, 0.94, 1.26]**
sitzt	[0.9, -0.3, 0.9, 1.0]	[0.58, 1.09, 0.86, 0.24]	**[1.48, 0.79, 1.76, 1.24]**
auf	[0.6, -1.0, 0.3, 1.4]	[0.38, 1.14, 0.78, 0.17]	**[0.98, 0.14, 1.08, 1.57]**
der	[0.1, -0.6, 0.0, 1.1]	[0.26, 1.12, 0.73, 0.13]	**[0.36, 0.52, 0.73, 1.23]**
Matte	[-1.0, 0.3, 0.1, 1.9]	[0.27, 1.27, 0.56, 0.13]	**[-0.73, 1.57, 0.66, 2.03]**

Layer Norm (Katze als Beispiel)

$$\text{LayerNorm}(x) = (x - \mu) / \sigma$$

μ = Mittelwert über alle Dimensionen, σ = Standardabweichung

Katze Summe = [2.06, 2.55, 0.94, 1.26]

μ = (2.06 + 2.55 + 0.94 + 1.26) / 4 = **1.70**

σ = √(((2.06-1.70)² + (2.55-1.70)² + (0.94-1.70)² + (1.26-1.70)²) / 4) = √0.406 = **0.64**

Normalisiert = [(2.06-1.70)/0.64, (2.55-1.70)/0.64, (0.94-1.70)/0.64, (1.26-1.70)/0.64] = **[0.56, 1.33, -1.19, -0.69]**

Token	μ	σ	Nach LayerNorm [d_1, d_2, d_3, d_4]
Die	1.55	0.56	**[0.80, 1.16, -1.16, -0.80]**
Katze	1.70	0.64	**[0.56, 1.33, -1.19, -0.69]**
sitzt	1.32	0.35	**[0.46, -1.51, 1.26, -0.23]**
auf	0.94	0.52	**[0.08, -1.54, 0.27, 1.21]**
der	0.71	0.32	**[-1.09, -0.59, 0.06, 1.63]**
Matte	0.88	0.98	**[-1.64, 0.70, -0.22, 1.17]**

6 Feed-Forward Network

$$\text{FFN}(z) = \text{ReLU}(z \cdot W_1 + b_1) \cdot W_2 + b_2$$

W_1: 4×8, W_2: 8×4, $b_1 = b_2 = 0$, ReLU(x) = max(0, x)

W_1 (4×8) — Aufblasen

	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8
d_1	1	0	-1	0	1	0	0	-1
d_2	0	1	0	-1	0	1	0	0

d_3	0	0	1	0	0	0	1	0
d_4	0	0	0	1	0	0	-1	1

W_2 (8×4) — Komprimieren

	d_1	d_2	d_3	d_4
h_1	1	0	0	0
h_2	0	1	0	0
h_3	0	0	0	0
h_4	0	0	0	0
h_5	0	0	1	0
h_6	0	0	0	1
h_7	0	0	0	0
h_8	0	0	0	0

FFN Berechnung (Katze)

```
Input = [0.56, 1.33, -1.19, -0.69]

×W₁ = [0.56, 1.33, -1.75, -2.02, 0.56, 1.33, -0.50, -1.25]
ReLU = [0.56, 1.33, 0, 0, 0.56, 1.33, 0, 0]
×W₂ = [0.56, 1.33, 0.56, 1.33]
```

7 Add & Layer Norm (2. Mal)

$$\text{BlockOutput} = \text{LayerNorm}(Z + \text{FFN}(Z))$$

```
Katze: Z + FFN(Z) = [0.56, 1.33, -1.19, -0.69] + [0.56, 1.33, 0.56, 1.33] = [1.12, 2.66, -0.63, 0.64]
μ = 0.95, σ = 1.18 → LayerNorm = [0.14, 1.45, -1.34, -0.26]
```

Block-Output für alle Tokens (nach 1 Block):

Token	d_1	d_2	d_3	d_4
Die	0.17	1.46	-1.34	-0.29
Katze	0.14	1.45	-1.34	-0.26
sitzt	0.25	-1.05	1.25	-0.44
auf	-0.15	-1.23	0.12	1.27
der	-1.12	-0.27	-0.07	1.46
Matte	-1.52	0.84	-0.18	0.86

In einem echten Transformer gehen diese Vektoren in Block 2, dann Block 3, ... bis Block 12. Wir nehmen den Output direkt für die Output-Schicht.

8 Output-Schicht: Logits → Softmax → nächstes Token

$$\text{Logit}_i = \text{VektorLetztes} \cdot \text{Embedding}_i \quad \text{(Skalarprodukt)}$$

Nur der letzte Token-Vektor (Matte) wird verwendet

Vektor von „Matte" nach Block 1: **[-1.52, 0.84, -0.18, 0.86]**

Token	Embedding	Skalarprodukt	Logit
Die	[0.9, 0.1, 0.0, 0.1]	(-1.52×0.9)+(0.84×0.1)+(-0.18×0.0)+(0.86×0.1)	**-1.20**
Katze	[0.0, 0.9, 0.1, 0.2]	(0)+(0.84×0.9)+(-0.18×0.1)+(0.86×0.2)	**0.91**
sitzt	[0.0, 0.1, 0.9, 0.0]	(0)+(0.84×0.1)+(-0.18×0.9)+(0)	**-0.08**
auf	[0.5, 0.0, 0.3, 0.4]	(-1.52×0.5)+(0)+(-0.18×0.3)+(0.86×0.4)	**-0.47**
der	[0.9, 0.1, 0.0, 0.1]	(-1.52×0.9)+(0.84×0.1)+(0)+(0.86×0.1)	**-1.20**
Matte	[0.0, 0.0, 0.0, 0.9]	(0)+(0)+(0)+(0.86×0.9)	**0.77**

Softmax (T = 1.0)

$$P(\text{token}_i) = e^{\text{logit}_i / T} / \Sigma\, e^{\text{logit}_j / T}$$

Token	Logit	e^{logit}	Wahrscheinlichkeit
Die	-1.20	0.301	4.6%
Katze	**0.91**	**2.484**	**37.9%**
sitzt	-0.08	0.923	14.1%
auf	-0.47	0.625	9.5%
der	-1.20	0.301	4.6%
Matte	0.77	2.160	33.0%
Summe		**6.794**	**≈ 100%**

Vorhersage: „Katze" (37.9%) — gefolgt von „Matte" (33.0%)
Der Transformer sagt nach nur 1 Block bereits ein inhaltlich passendes Token vorher.

Mit Temperature

Token	T = 0.5 (fokussiert)	T = 1.0 (normal)	T = 2.0 (kreativ)
Die	1.4%	4.6%	8.7%
Katze	**52.7%**	**37.9%**	**24.7%**
sitzt	7.3%	14.1%	16.3%
auf	3.3%	9.5%	13.1%
der	1.4%	4.6%	8.7%
Matte	39.2%	33.0%	23.7%

Σ Alle Formeln auf einen Blick

#	Schicht	Formel	Dimensionen
1	**Tokenizer**	`token_ids = tokenize(text)`	→ (n,)
2	**Embedding**	`E = EmbTable[token_ids]`	→ (n, d)
3	**Pos. Encoding**	`X = E + PE`	→ (n, d)
4a	**Q, K, V**	$Q = X \cdot W_Q$, $K = X \cdot W_K$, $V = X \cdot W_V$	→ (n, d_k) je
4b	**Attention Score**	$S = (Q \cdot K^T) / \sqrt{d_k}$	→ (n, n)
4c	**Mask**	$S[i,j] = -\infty$ wenn $j > i$	→ (n, n)
4d	**Softmax**	$A = \text{softmax}(S) = e^S / \Sigma e^S$	→ (n, n)
4e	**Attention Out**	$\text{Head} = A \cdot V$	→ (n, d_k)
4f	**Multi-Head**	$MH = \text{Concat}(\text{Head}_1, ..., \text{Head}_h) \cdot W_O$	→ (n, d)
5a	**Add & Norm 1**	`Z = LayerNorm(X + MH)`	→ (n, d)
6a	**FFN**	$FFN = \text{ReLU}(Z \cdot W_1 + b_1) \cdot W_2 + b_2$	d → d_{ff} → d
5b	**Add & Norm 2**	`Out = LayerNorm(Z + FFN(Z))`	→ (n, d)
	↑ Schritte 4–6 wiederholen sich L Mal (12× bei GPT-2) ↑		
7	**Logits**	$\text{logits} = \text{Out}[-1] \cdot \text{EmbTable}^T$	→ (V,)
8	**Wahrsch.**	`P = softmax(logits / T)`	→ (V,)
9	**Auswahl**	`next_token = sample(P)`	→ 1 Token

Parameterzählung

Komponente	Formel	Unser Modell	GPT-2 Small
Embedding-Tabelle	V × d	6 × 4 = 24	50.257 × 768 = 38.6M
Positional Encoding	berechnet	0	1.024 × 768 = 0.8M *
W_Q, W_K, W_V (alle Heads)	3 × d × d	3 × 4 × 4 = 48	3 × 768² = 1.8M
W_O	d × d	4 × 4 = 16	768² = 0.6M
FFN (W_1 + W_2)	$d \times d_{ff} + d_{ff} \times d$	4×8 + 8×4 = 64	2 × 768 × 3072 = 4.7M
LayerNorm (γ, β)	2 × 2 × d	16	2 × 2 × 768 = 3K
Pro Block		**144**	**~7.1M**
× L Blöcke		**× 1 = 144**	**× 12 = 85M**
+ Embedding			**+ 39.4M**
Gesamt		**~168**	**~124M**

* GPT-2 nutzt gelernte Positional Embeddings statt Sinus/Cosinus

Komplette Mathematik eines Transformer-Blocks — berechenbar auf Papier.

David & Claude · April 2026