Schritt 4b: Masked Attention & Multi-Head

Tokenizer ✓ → Embedding ✓ → Pos. Encoding ✓ → **Masked Multi-Head Attention** → Add & Norm → FFN

Warum eine Maske? — Das Schummel-Problem

Erlebnispark: *Stell dir vor, GPT soll den Satz vervollständigen: „Die Katze sitzt auf der ___"*

Wenn „Katze" bei der Attention auf **alle** *Wörter schauen darf — auch auf „Matte" am Ende — dann kennt es die Antwort schon.* **Das ist Schummeln!**

Die Maske ist ein Vorhang: Jedes Token darf nur nach **links** *schauen — auf sich selbst und auf alles was vorher kam. Was danach kommt ist verdeckt.*

Die Maske: Ein Dreieck

↓ sieht →	Die	Katze	sitzt	auf	der	Matte
Die	✓	-∞	-∞	-∞	-∞	-∞
Katze	✓	✓	-∞	-∞	-∞	-∞
sitzt	✓	✓	✓	-∞	-∞	-∞
auf	✓	✓	✓	✓	-∞	-∞
der	✓	✓	✓	✓	✓	-∞
Matte	✓	✓	✓	✓	✓	✓

So funktioniert die Maske:

1. Berechne alle Scores normal
2. Setze verbotene Scores auf **-∞**
3. Nach Softmax: $e^{-\infty}$ = **0**

Die maskierten Tokens werden unsichtbar — als wären sie nicht da.

Beachte: „Die" (Position 0) kann **nur sich selbst sehen**! Ihr gesamtes Attention-Gewicht (100%) geht an sich selbst. „Matte" (Position 5) kann alle sehen — sie hat die meiste Information.

Katze mit Maske — Neuberechnung

Erlebnispark: *Du bist wieder „Katze" (Position 1). Du setzt die Brille auf — aber diesmal hängt ein* **Vorhang** *im Raum. Du kannst nur nach links schauen: Du siehst* **„Die"** *und* **dich selbst**. *Alle anderen sind hinter dem Vorhang.*

Aus Schritt 4a kennen wir die Scores (bevor Softmax):

Katze →	Die	Katze	sitzt	auf	der	Matte
Score (vorher)	1.54	1.76	2.12	2.20	1.54	2.74
Nach Maske	1.54	1.76	-∞	-∞	-∞	-∞

Skalieren (÷ √2 = 1.41):

```
Die:   1.54 / 1.41 = 1.09
Katze: 1.76 / 1.41 = 1.25
Rest: -∞ (bleibt -∞)
```

Softmax (nur über die sichtbaren Tokens!):

```
e^1.09 = 2.97
e^1.25 = 3.49
e^-∞  = 0   (vier Mal)

Summe = 2.97 + 3.49 + 0 + 0 + 0 + 0 = 6.46

Die:   2.97 / 6.46 = 0.46   (46%)
Katze: 3.49 / 6.46 = 0.54   (54%)
sitzt bis Matte: 0.00 (0%)
```

Attention-Gewichte visualisiert:

sitzt–Matte 0.00

Gewichtete Summe der Values:

$$\text{Output} = 0.46 \times V_{\text{Die}} + 0.54 \times V_{\text{Katze}}$$

$$= 0.46 \times [1.1, 1.1] + 0.54 \times [1.4, 1.2]$$

$$= [0.506, 0.506] + [0.756, 0.648]$$

$$= \mathbf{[1.26, 1.15]}$$

Vergleich: Mit vs. Ohne Maske

	Ohne Maske (Encoder)	Mit Maske (Decoder/GPT)
Sichtbare Tokens	Alle 6	Nur Die + Katze
Stärkstes Gewicht	Matte (27%)	Katze selbst (54%)
Output	[0.10, 1.38]	**[1.26, 1.15]**

Komplett anderes Ergebnis!
Ohne Maske wusste Katze über die Matte Bescheid. Mit Maske weiß Katze nur: *„Vor mir steht ein Artikel. Ich bin ein Substantiv."* Die Matte kennt sie noch nicht — die kommt erst später im Satz.

Multi-Head Attention: Zwei Räume gleichzeitig

Erlebnispark: *Du wirst in* **2 Räume gleichzeitig** *geschickt. Nicht komplett kopiert — du wirst* **in Scheiben geschnitten**:

Raum 1 bekommt deine ersten 2 Dimensionen (d_1, d_2).
Raum 2 bekommt deine letzten 2 Dimensionen (d_3, d_4).

Jeder Raum hat eine **andere Brille** *— und sieht deshalb andere Zusammenhänge.*

Bei uns: 4 Dimensionen ÷ 2 Heads = 2 Dimensionen pro Head
Bei GPT-2: 768 Dimensionen ÷ 12 Heads = 64 Dimensionen pro Head

Jeder Head hat **eigene** W_Q, W_K, W_V-Matrizen. Dadurch stellt jeder Head eine andere Frage an den Satz.

Head 1 — „Wer bin ich und was bieten die anderen?"

Head 1 (Raum 1)

W_{Q1}	
1	0
0	1
0	0
0	0

Q = [d_1, d_2]

W_{K1}	
0	0
0	0
1	0
0	1

K = [d_3, d_4]

W_{V1}	
0	0
1	0
0	0
0	1

V = [d_2, d_4]

Das ist genau das, was wir schon berechnet haben!
Output Head 1 für Katze (mit Maske): [1.26, 1.15]

Head 2 — „Was ist mein Kontext und wer passt dazu?"

Head 2 (Raum 2)

W_{Q2}		W_{K2}		W_{V2}	
0	0	1	0	1	0
0	0	0	1	0	0
1	0	0	0	0	1
0	1	0	0	0	0
Q = [d_3, d_4]		K = [d_1, d_2]		V = [d_1, d_3]	

Head 2 stellt die umgekehrte Frage: Q nimmt d_3, d_4 (Kontext-Dimensionen) und sucht passende Keys in d_1, d_2 (Bedeutungs-Dimensionen). Eine andere Perspektive!

Q, K, V für Head 2 berechnen (nur Katze und Die — wegen Maske):

Token	Input [d_1, d_2, d_3, d_4]	Q_2 [d_3, d_4]	K_2 [d_1, d_2]	V_2 [d_1, d_3]
Die	[0.9, 1.1, 0.0, 1.1]	[0.0, 1.1]	[0.9, 1.1]	[0.9, 0.0]
Katze	[0.8, 1.4, 0.1, 1.2]	[0.1, 1.2]	[0.8, 1.4]	[0.8, 0.1]

Scores für Katze in Head 2 (mit Maske):

$Q_{2\text{Katze}}$ = [0.1, 1.2]

Katze → Die: [0.1, 1.2] · [0.9, 1.1] = (0.1×0.9) + (1.2×1.1) = 0.09 + 1.32 = **1.41**

Katze → Katze: [0.1, 1.2] · [0.8, 1.4] = (0.1×0.8) + (1.2×1.4) = 0.08 + 1.68 = **1.76**

Skalieren + Softmax:

Skaliert: Die = 1.41/1.41 = **1.00**, Katze = 1.76/1.41 = **1.25**

$e^{1.00}$ = 2.72, $e^{1.25}$ = 3.49, Summe = 6.21

Die: 2.72 / 6.21 = 0.44 (44%)

Katze: 3.49 / 6.21 = 0.56 (56%)

Gewichtete Summe:

$$\text{Output} = 0.44 \times V_{₂\text{Die}} + 0.56 \times V_{₂\text{Katze}}$$

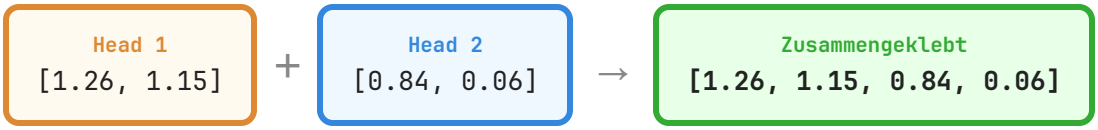
$$= 0.44 \times [0.9, 0.0] + 0.56 \times [0.8, 0.1]$$

$$= [0.396, 0.000] + [0.448, 0.056]$$

$$= \mathbf{[0.84, 0.06]}$$

Die Heads zusammenkleben: Concatenation

Erlebnispark: *Du verlässt beide Räume gleichzeitig. Raum 1 gibt dir ein Paket mit 2 Zahlen. Raum 2 gibt dir ein Paket mit 2 Zahlen. Die Pakete werden* **zusammengeklebt** *— und du bist wieder* **4-dimensional***.*



Concatenation heißt einfach: die Ergebnisse hintereinander hängen.
2 Zahlen (Head 1) + 2 Zahlen (Head 2) = 4 Zahlen = zurück auf unsere Original-Dimension!

Bei GPT-2: 12 Heads × 64 = 768 — genau die Original-Dimension.

Output-Projektion: W_O

In der Praxis gibt es noch eine letzte Matrix $\mathbf{W_O}$ (4×4), die den zusammengeklebten Vektor nochmal transformiert. Für unser Papier-Modell verwenden wir die Identitätsmatrix (= keine Veränderung), damit die Zahlen übersichtlich bleiben.

Attention-Output für „Katze" = [1.26, 1.15, 0.84, 0.06]

Was hat jeder Head gelernt?

	Head 1	Head 2
Fragt nach (Q)	d_1, d_2 (Bedeutung: Artikel? Lebewesen?)	d_3, d_4 (Kontext: Aktion? Objekt?)
Sucht in (K)	d_3, d_4 der anderen	d_1, d_2 der anderen
Gewichte	Die: 46%, Katze: 54%	Die: 44%, Katze: 56%

Nimmt mit (V)	d_2, d_4 → [1.26, 1.15]	d_1, d_3 → [0.84, 0.06]

Verschiedene Heads, verschiedene Perspektiven!
Beide Heads schauen auf Die und Katze (wegen der Maske), aber sie extrahieren **verschiedene Informationen**: Head 1 sammelt Lebewesen/Objekt-Info, Head 2 sammelt Bedeutungs/Kontext-Info.

In einem trainierten GPT-2 mit 12 Heads schaut ein Head auf Syntax, ein anderer auf Korreferenzen, ein dritter auf Nachbar-Tokens, usw.

Übung: Berechne Multi-Head Attention für „sitzt" (Position 2)

„sitzt" kann sehen: **Die, Katze, sich selbst** (3 Tokens sichtbar!).
$Input_{sitzt}$ = [0.9, -0.3, 0.9, 1.0]

Head 1:

✏️ Selbst rechnen

$Q1_{sitzt}$ = [d_1, d_2] = [0.9, -0.3]

sitzt →	K_1-Vektor [d_3,d_4]	Score	÷ 1.41
Die	[0.0, 1.1]		
Katze	[0.1, 1.2]		
sitzt	[0.9, 1.0]		

Softmax: e^___ = ___, e^___ = ___, e^___ = ___ → Summe = ___

Gewichte: Die = ___, Katze = ___, sitzt = ___

V_1: Die=[1.1, 1.1], Katze=[1.4, 1.2], sitzt=[-0.3, 1.0]

Output Head 1 = ___ × [1.1,1.1] + ___ × [1.4,1.2] + ___ × [-0.3,1.0] = [___, ___]

Head 2:

✏️ Selbst rechnen

$Q2_{sitzt}$ = [d_3, d_4] = [0.9, 1.0]

sitzt →	K_2-Vektor [d_1,d_2]	Score	÷ 1.41
Die	[0.9, 1.1]		
Katze	[0.8, 1.4]		

sitzt	[0.9, -0.3]		

Softmax: Gewichte: Die = ___, Katze = ___, sitzt = ___

V_2: Die=[0.9, 0.0], Katze=[0.8, 0.1], sitzt=[0.9, 0.9]

Output Head 2 = [___, ___]

Zusammenkleben:

✏️ Selbst rechnen

Attention-Output$_{sitzt}$ = Head 1 [___, ___] + Head 2 [___, ___] = [___, ___, ___, ___]

Zusammenfassung: Masked Multi-Head Attention

Schritt	Was passiert	Bei uns
1. Aufteilen	Input wird an die Heads verteilt. Jeder Head hat eigene W_Q, W_K, W_V	2 Heads × 2D
2. Q, K, V	Jeder Head berechnet eigene Queries, Keys, Values	2D-Vektoren
3. Scores	Q · K — aber maskierte Positionen = -∞	Dreieck-Maske
4. Softmax	$e^{-\infty} = 0$ → zukünftige Tokens unsichtbar	Nur sichtbare zählen
5. Gewichtete Summe	Values der sichtbaren Tokens einsammeln	→ 2D Output pro Head
6. Zusammenkleben	Alle Head-Outputs concatenieren	2 × 2D = 4D
7. W_O	Output-Projektion (letzte Transformation)	4D → 4D

Unsere Ergebnisse (mit Maske):

Token	Sichtbare Tokens	Attention-Output
Die	nur sich selbst	berechne selbst!
Katze	Die, Katze	**[1.26, 1.15, 0.84, 0.06]**
sitzt	Die, Katze, sitzt	deine Übung!
auf	Die, Katze, sitzt, auf	berechne selbst!
der	Die, Katze, sitzt, auf, der	berechne selbst!
Matte	alle	berechne selbst!

Erlebnispark — Zwischenstand: *Jedes Token hat jetzt 2 Räume durchlaufen, mit Vorhang (Maske). Jedes hat Information von den Tokens aufgesammelt die es sehen durfte.*

Aber wir sind noch nicht fertig! *Als Nächstes kommt:* **Add & Normalize** *(das Original*

zurückaddieren und die Werte stabilisieren) und dann das **Feed-Forward-Netzwerk** *(wo jedes Token für sich allein nachdenkt).*