# Attention

## Autor und Herausgeber

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland (ORCID 0000-0001-7538-0990)
Editors: Konstantina Olioumtsevits (ORCID 0000-0003-2697-8659), Ekkehard Schnoor (ORCID 0000-0001-8252-1499), Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

## Zusammenfassung

Attention ist ein Mechanismus, der die Abhängigkeiten zwischen den Tokens modelliert, aus denen ein Datenpunkt besteht, wie etwa den Wörtern in einem Satz oder den Bildausschnitten eines Bildes. Attention stellt die Beziehung zwischen zwei Tokens durch eine parametrisierte Gewichts-Funktion dar, deren Modellparameter aus einem Trainingsdatensatz gelernt werden. Ein Attention-Kopf berechnet anschließend für jedes Token eine neue Vektordarstellung als gewichtete Kombination der Value-Vektoren aller Tokens. Wie die Gewichts-Funktion werden auch die Value-Vektoren gelernt. Ein Attention-Kopf fungiert als differenzierbarer assoziativer Speicher: Ein Token nutzt seinen Query-Vektor, um über die Key-Vektoren der anderen Tokens die für es relevantesten Values abzurufen. Attention-Köpfe stellen Abhängigkeiten zwischen Tokens unabhängig von deren Positionen innerhalb eines Datenpunkts dar und sind eine Kernkomponente moderner große Sprachmodelle (LLM).

## Definition

Ein Satz besteht aus Wörtern, ein Bild aus Bildausschnitten und ein Netzwerk aus Knoten: Die Datenpunkte mancher maschinelles Lernen (ML)-Anwendungen setzen sich aus kleineren Einheiten zusammen, die als Tokens bezeichnet werden. Im Allgemeinen sind die Tokens, die einen einzelnen Datenpunkt bilden, nicht unabhängig voneinander. Stattdessen hängt die Bedeutung eines Tokens von einigen anderen Tokens desselben Datenpunkts ab, die weit von ihm entfernt liegen können. Der Attention-Mechanismus ist ein Baustein künstlicher neuronaler Netze, der solche Abhängigkeiten zwischen Tokens unabhängig von ihrem Abstand innerhalb eines Datenpunkts darstellt (Vaswani et al., 2017, Sect. 4), und er ist eine Kernkomponente der auf dem Transformer aufbauenden große Sprachmodelle (LLM) (Brown et al., 2020).

Wahrscheinlichkeitsmodelle stellen solche Abhängigkeiten dar und analysieren sie (Blei et al., 2003). Attention-Mechanismen stellen diese Abhängigkeiten stattdessen direkt dar, ohne eine Wahrscheinlichkeitsverteilung über die Tokens anzugeben. Sie stellen die Beziehung zwischen einem Token mit Index $i$ und einem Token mit Index $i'$ durch ein Attention-Gewicht $f^{(\mathbf{w})}(i, i')$ dar, eine parametrisierte Funktion, deren Modellparameter $\mathbf{w}$ aus einem Trainingsdatensatz gelernt werden. Das Attention-Gewicht misst, wie stark Token $i$ seine Aufmerk-

samkeit auf Token $i'$ richtet. Praktische Attention-Mechanismen unterscheiden sich in der Wahl der Funktion $f^{(\mathbf{w})}(i, i')$ und in der zum Lernen von $\mathbf{w}$ verwendeten Variante der empirischen Risikominimierung. Die am weitesten verbreitete Wahl, die im Folgenden beschrieben wird, ist die skalierte Skalarprodukt-Attention (engl. *scaled dot-product attention*) (Vaswani et al., 2017).

Die skalierte Skalarprodukt-Attention leitet aus dem Einbettung-Vektor $\mathbf{x}^{(i)} \in \mathbb{R}^d$ jedes Tokens $i$ über gelernte Matrizen drei Vektoren ab. Diese sind ein Query (-Vektor) $\mathbf{q}^{(i)} = \mathbf{W}_Q \, \mathbf{x}^{(i)}$, ein Key (-Vektor) $\mathbf{k}^{(i)} = \mathbf{W}_K \, \mathbf{x}^{(i)}$ und ein Value (-Vektor) $\mathbf{v}^{(i)} = \mathbf{W}_V \, \mathbf{x}^{(i)}$, wobei die Matrizen

$$\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d \times d}$$

als Modellparameter während des Trainings gelernt werden. Der Query eines Tokens bestimmt, zu welchen Keys er passt, der Key eines Tokens bestimmt, welche Queries zu ihm passen, und der Value trägt die Information, die aggregiert wird (Bishop and Bishop, 2024; Elhage et al., 2021).

Die Ausgabe des Attention-Mechanismus für Token $i$ ist eine gewichtete Summe der Values aller Tokens,

$$\mathbf{z}^{(i)} = \sum_{i'=1}^{n} \alpha_{i,i'} \, \mathbf{v}^{(i')}.$$

Hierbei bezeichnet $n$ die Anzahl der Tokens des Datenpunkts. Der Koeffizient $\alpha_{i,i'}$ ist das Attention-Gewicht, d. h. die skalierte Skalarproduktform von $f^{(\mathbf{w})}(i, i')$. Die Attention-Gewichte werden in zwei Schritten aus den Queries und Keys berechnet. Zunächst ist der Attention-Score zwischen den Tokens $i$ und $i'$ gegeben durch

$$s_{i,i'} = \frac{\left(\mathbf{q}^{(i)}\right)^\top \mathbf{k}^{(i')}}{\sqrt{d}}.$$

Das Skalarprodukt (auch Punktprodukt genannt) $\left(\mathbf{q}^{(i)}\right)^\top \mathbf{k}^{(i')}$ misst, wie gut der Key von Token $i'$ zum Query von Token $i$ passt. Anschließend ergibt sich das Attention-Gewicht $\alpha_{i,i'}$ durch Anwenden der Softmax-Funktion auf die Scores $s_{i,i'}$ über alle Tokens $i'$ hinweg:

$$\alpha_{i,i'} = \frac{\exp(s_{i,i'})}{\sum_{j=1}^{n} \exp(s_{i,j})}.$$

Die Softmax-Funktion normiert die Scores. Die Attention-Gewichte $\alpha_{i,i'}$ sind nichtnegativ und summieren sich über $i' = 1, \ldots, n$ zu eins, sodass sie eine Wahrscheinlichkeitsverteilung über die Tokens bilden. Das Teilen des Skalarprodukts durch $\sqrt{d}$ im Attention-Score wirkt seinem Anwachsen mit der Dimension $d$ entgegen. Der Score $\left(\mathbf{q}^{(i)}\right)^\top \mathbf{k}^{(i')}$ ist eine Summe von $d$ Termen. Ohne die Skalierung würde ein großes $d$ die Softmax-Funktion daher in einen Bereich treiben, in dem ihre Gradienten nahezu verschwinden (Vaswani et al., 2017, Sect. 3.2.1). Abb. 1 veranschaulicht die skalierte Skalarprodukt-Attention anhand eines Satzes aus acht Tokens.

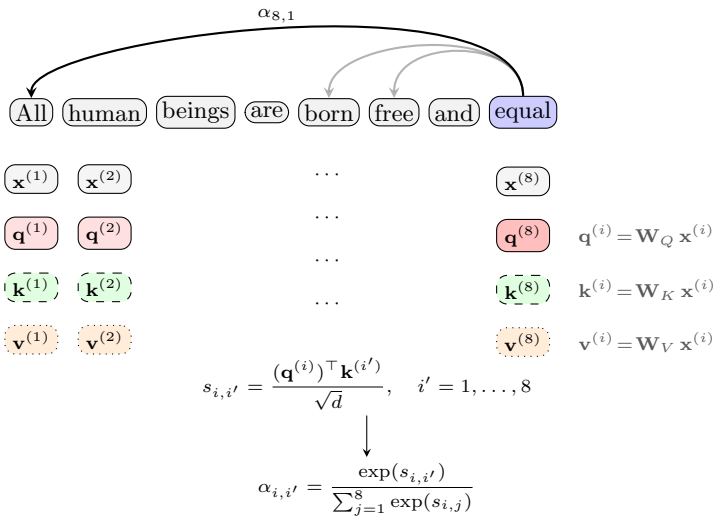



Abbildung 1: Skalierte Skalarprodukt-Attention für den Satz „All human beings are born free and equal.“ Jedes Token-Einbettung $\mathbf{x}^{(i)}$ wird durch gelernte Matrizen $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ auf Query-, Key- und Value-Vektoren abgebildet (Kästchen mit durchgezogenem, gestricheltem und gepunktetem Rand). Zunächst wird der Attention-Score $s_{i,i'}$ als skaliertes Skalarprodukt von Query $i$ und Key $i'$ berechnet. Anschließend normiert die Softmax-Funktion diese Scores zu Attention-Gewichten $\alpha_{i,i'}$, die sich über alle Positionen hinweg zu eins summieren. Bögen am oberen Rand veranschaulichen, wie Token 8 („equal“) seine Aufmerksamkeit auf frühere Tokens richtet; dickere Bögen zeigen höhere Attention-Gewichte an (Vaswani et al., 2017). Das Token „equal“ und sein Query $\mathbf{q}^{(8)}$ sind durch eine dunklere Füllung hervorgehoben: Die dargestellten Gewichte $\alpha_{8,i'}$ beziehen den Query von Token 8, aber die Keys und Values aller Tokens ein

Ein Attention-Kopf geht von einem einzelnen Token zur gesamten Folge von Tokens über. Er ist eine parametrisierte Funktion $h^{(\mathbf{w})}$, die die Eingabe-Einbettungen auf die Ausgabe-Vektoren abbildet,

$$h^{(\mathbf{w})}\big(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\big) = \big(\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(n)}\big),$$

wobei die Modellparameter $\mathbf{w}$ die drei Matrizen $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ umfassen. Fasst man die Ausgabe-Vektoren als Zeilen einer Matrix $\mathbf{Z}$ und die Queries, Keys und Values als Zeilen von $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ zusammen, so berechnet $h^{(\mathbf{w})}$ diese in der kompakten Form

$$\mathbf{Z} = \mathrm{softmax}\!\left(\frac{\mathbf{Q}\,\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V}.$$

Hierbei wird die Softmax-Funktion unabhängig auf jede Zeile von $\mathbf{Q}\,\mathbf{K}^\top/\sqrt{d}$ angewendet, sodass jede Zeile der resultierenden Matrix eine Wahrscheinlich-

keitsverteilung über die $n$ Tokens ist. Da die Queries, Keys und Values alle aus derselben Folge von Tokens abgeleitet werden, wird diese Konstruktion Self-Attention (Selbstaufmerksamkeit) genannt.

In der Praxis arbeiten mehrere Attention-Köpfe, jeder mit seinen eigenen gelernten Matrizen $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$, parallel; dies wird Multi-Head-Attention genannt, und jeder Kopf kann sich auf eine andere Beziehung zwischen Tokens spezialisieren (Vaswani et al., 2017).

Attention lässt sich als differenzierbarer assoziativer Speicher interpretieren, d. h. als inhaltsadressierbarer Speicher (Ramsauer et al., 2021, Sect. 2, Eq. (3) and Eq. (10)): Die Keys $\mathbf{k}^{(i')}$ fungieren als Adressen, die Values $\mathbf{v}^{(i')}$ als gespeicherte Inhalte, und ein Token $i$ durchsucht die anderen Tokens mit seinem Query $\mathbf{q}^{(i)}$. Das Skalarprodukt $\left(\mathbf{q}^{(i)}\right)^\top \mathbf{k}^{(i')}$ misst, wie gut die Adresse $i'$ zum Query passt, die Softmax-Funktion wandelt die Übereinstimmungswerte in Abrufgewichte um, und die Ausgabe $\mathbf{z}^{(i)}$ ist der abgerufene Inhalt.

Eine klassische Wörterbuch-Datenstruktur oder Hashtabelle gibt den einzelnen Wert zurück, der an einer exakt übereinstimmenden Adresse gespeichert ist (Cormen et al., 2022, Ch. 11). Attention gibt stattdessen ein gewichtetes Mittel zurück. Darüber hinaus ist ihre Ausgabe $\mathbf{z}^{(i)}$ eine differenzierbare Funktion der Matrizen $\mathbf{W}_Q$, $\mathbf{W}_K$ und $\mathbf{W}_V$. Insbesondere können die Queries, Keys und Values selbst über gradientenbasierte Verfahren zur Lösung der empirischen Risikominimierung gelernt werden. Dasselbe Abrufprinzip, mit einem gewichteten Mittel anstelle eines einzelnen besten Treffers, wird von k-nächste Nachbarn (k-NN)-Verfahren verwendet. Die Berechnung aller paarweisen Scores $s_{i,i'}$ erfordert $n^2$ Query-Key-Skalarprodukte, eines für jedes geordnete Paar von Tokens. Das Abrufprinzip leitet auch effiziente Implementierungen an: Locality-Sensitive Hashing beschränkt jeden Query auf nahegelegene Keys, wodurch sich der Aufwand eines Attention-Kopfes von $n^2$ Query-Key-Vergleichen auf ein konstantes Vielfaches von $n \ln n$ reduziert, wobei $n$ wiederum die Anzahl der Tokens ist (Kitaev et al., 2020, Sect. 1).

Da $\mathbf{q}^{(i)}$ und $\mathbf{k}^{(i')}$ aus den unterschiedlichen Matrizen $\mathbf{W}_Q$ und $\mathbf{W}_K$ gewonnen werden, sind die Attention-Gewichte *gerichtet*: Im Allgemeinen gilt $\alpha_{i,i'} \neq \alpha_{i',i}$, sodass eine starke Aufmerksamkeit von Token $i$ auf Token $i'$ nicht den umgekehrten Fall impliziert.

Abb. 2 veranschaulicht die Attention-Gewichte $\alpha_{i,i'}$, die ein einzelner Kopf auf Sätzen aus der Allgemeinen Erklärung der Menschenrechte (United Nations General Assembly, 1948) lernt. Trainiert darauf, jedes maskierte Token aus den übrigen zu rekonstruieren, weist der Kopf vielen Query-Tokens ein auf wenige Keys konzentriertes Gewicht zu, anstatt es gleichmäßig zu verteilen, was die Interpretation als assoziativer Speicher auf diesem kleinen Korpus veranschaulicht. Die Heatmap zeigt auch die oben besprochene gerichtete Natur der Gewichte: Das Query-Token „rights“ legt sein volles Gewicht auf das Key-Token „beings“ ($\alpha_{12,3} = 1.0$), während das Query-Token „beings“ kein Gewicht auf das Key-Token „rights“ legt ($\alpha_{3,12} = 0.0$).

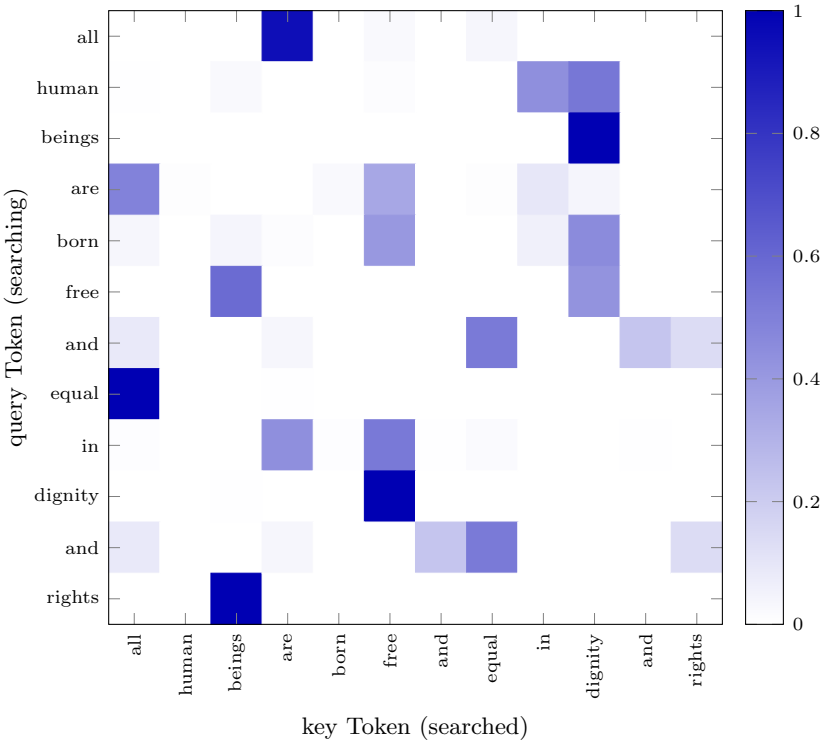



Abbildung 2: Attention-Gewichte $\alpha_{i,i'}$ eines Self-Attention-Heads für den Satz „All human beings are born free and equal in dignity and rights“ (Artikel 1 der Allgemeinen Erklärung der Menschenrechte, United Nations General Assembly, 1948). Zeile $i$ ist das Query-Token; Spalte $i'$ ist das gesuchte Key-Token. Dunklere Zellen tragen mehr Gewicht: Beispielsweise legt das Query-Token „all“ nahezu sein gesamtes Gewicht auf das Key-Token „are“ ($\alpha_{1,4} = 0.94$), und seine übrigen Gewichte liegen nahe null, da sich die Gewichte jeder Query zu eins summieren. Der Head wurde auf Sätzen aus der Allgemeinen Erklärung der Menschenrechte trainiert, indem jedes Token maskiert und aus den anderen rekonstruiert wurde, sodass die Query assoziierte Tokens über deren Keys abrufen muss. Daten erzeugt durch `pythondemos/attention.py`

Synonyme: Aufmerksamkeitsmechanismus, Attention-Mechanismus, Attention-Kopf.

Siehe auch: Token, Einbettung, Transformer, Softmax-Funktion, LLM, k-NN.

## Querverweise

## Literatur


1. Bishop and Bishop (2024). Deep Learning: Foundations and Concepts. Springer Nature, Cham, Switzerland.

2. Blei et al. (2003). Latent Dirichlet Allocation. J. Mach. Learn. Res., 3, pp. 993–1022.

3. Brown et al. (2020). Language Models are Few-Shot Learners. In: Adv. Neural Inf. Process. Syst., pp. 1877–1901.

4. Cormen et al. (2022). Introduction to Algorithms. 4th ed., MIT Press, Cambridge, MA, USA.

5. Elhage et al. (2021). A Mathematical Framework for Transformer Circuits. Transformer Circuits Thread.

6. United Nations General Assembly (1948). Universal Declaration of Human Rights (UDHR). United Nations, New York.

7. Kitaev et al. (2020). Reformer: The Efficient Transformer. In: Int. Conf. Learn. Represent. (ICLR).

8. Ramsauer et al. (2021). Hopfield networks is all you need. In: Int. Conf. Learn. Represent. (ICLR).

9. Vaswani et al. (2017). Attention is all you need. In: Adv. Neural Inf. Process. Syst., pp. 5998–6008.