Wie Computer mit Sprache umgehen (LLMs)

Kapitel online

Wie aus Sätzen Zahlen werden: Text in Häppchen zerlegen (Tokenization) und Wörter als Zahlenreihen ablegen (Word Embeddings) — die Grundlage jedes Sprachmodells.

Grundlagen · Kapitel 8 von 50 · ca. 35 Min. · 100 XP

Einfach gesagt: Dein Satz wird in Wortstücke zerlegt, jedes bekommt eine Nummer und dann einen Ort auf einer Bedeutungs-Landkarte aus hunderten Zahlen.

Stell dir vor: Du schreibst "Hallo Welt" in ChatGPT. Aber wie versteht die KI deine Worte? Computer können doch nur Zahlen! **Die Lösung: Tokenization und Word Embeddings!**

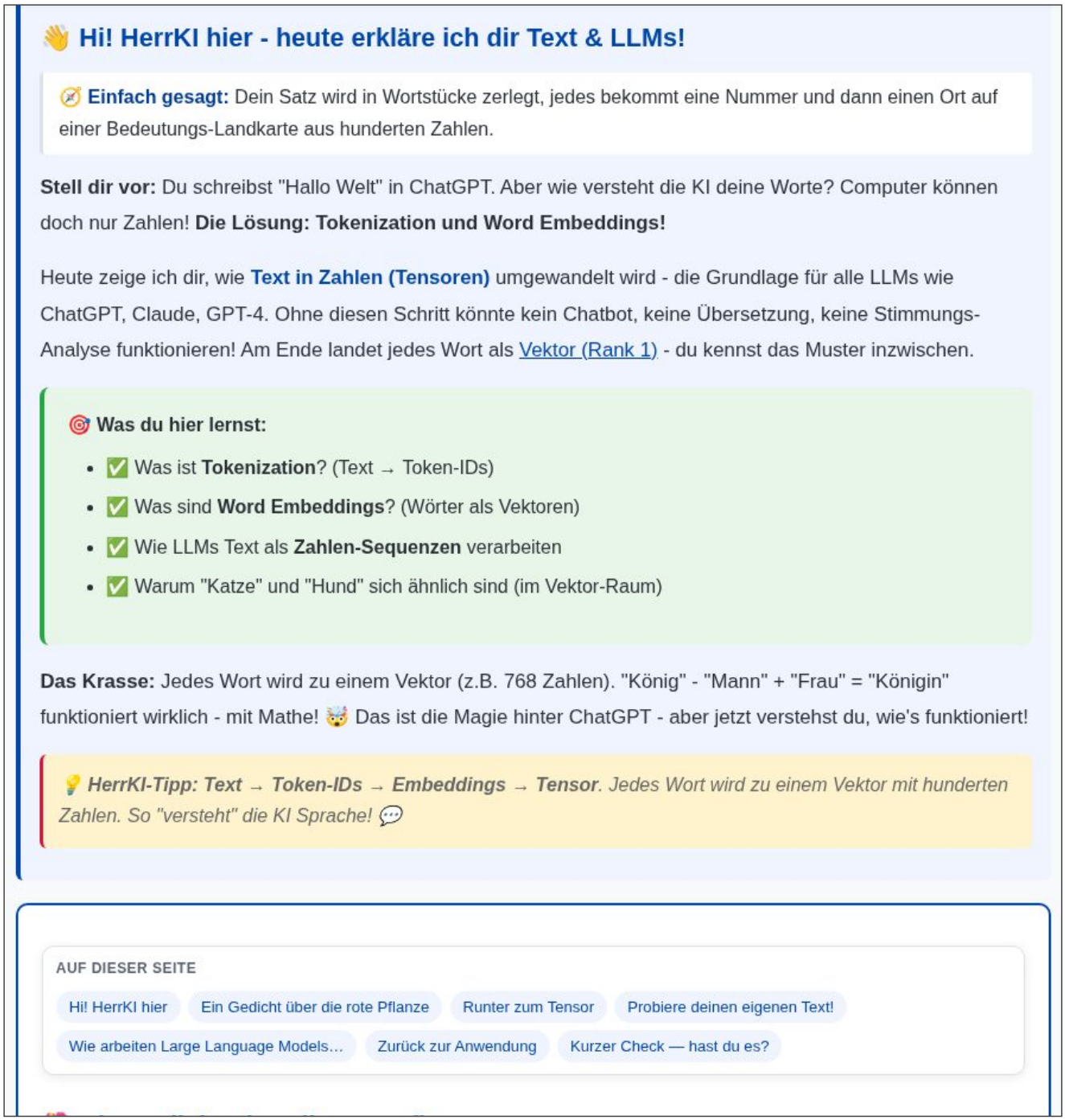
Heute zeige ich dir, wie **Text in Zahlen (Tensoren)** umgewandelt wird - die Grundlage für alle LLMs wie ChatGPT, Claude, GPT-4. Ohne diesen Schritt könnte kein Chatbot, keine Übersetzung, keine Stimmungs-Analyse funktionieren! Am Ende landet jedes Wort als Vektor (Rank 1) - du kennst das Muster inzwischen.

Das Krasse: Jedes Wort wird zu einem Vektor (z.B. 768 Zahlen). "König" - "Mann" + "Frau" = "Königin" funktioniert wirklich - mit Mathe! Das ist die Magie hinter ChatGPT - aber jetzt verstehst du, wie's funktioniert!

HerrKI-Tipp: Text → Token-IDs → Embeddings → Tensor. Jedes Wort wird zu einem Vektor mit hunderten Zahlen. So "versteht" die KI Sprache!

Am Gerät ausprobieren: herrki.de/040_text_und_llms#abschnitt-1

Direkt zum Modul



Interaktives Modul – am Gerät ausprobieren

Was du hier lernst

- Was ist **Tokenization**? (Text → Token-IDs)
- Was sind **Word Embeddings**? (Wörter als Vektoren)
- Wie LLMs Text als **Zahlen-Sequenzen** verarbeiten
- Warum "Katze" und "Hund" sich ähnlich sind (im Vektor-Raum)

Für den Unterricht

Einstieg (7 Min): Kapitel gemeinsam lesen, Alltagsbeispiel aufgreifen.

Erkunden am Gerät (21 Min): die interaktiven Module auf herrki.de/040_text_und_llms ausprobieren.

Sichern mit Quiz (7 Min): „Kurzer Check" besprechen, „Das nimmst du mit" festhalten.

Ein Gedicht über die rote Pflanze

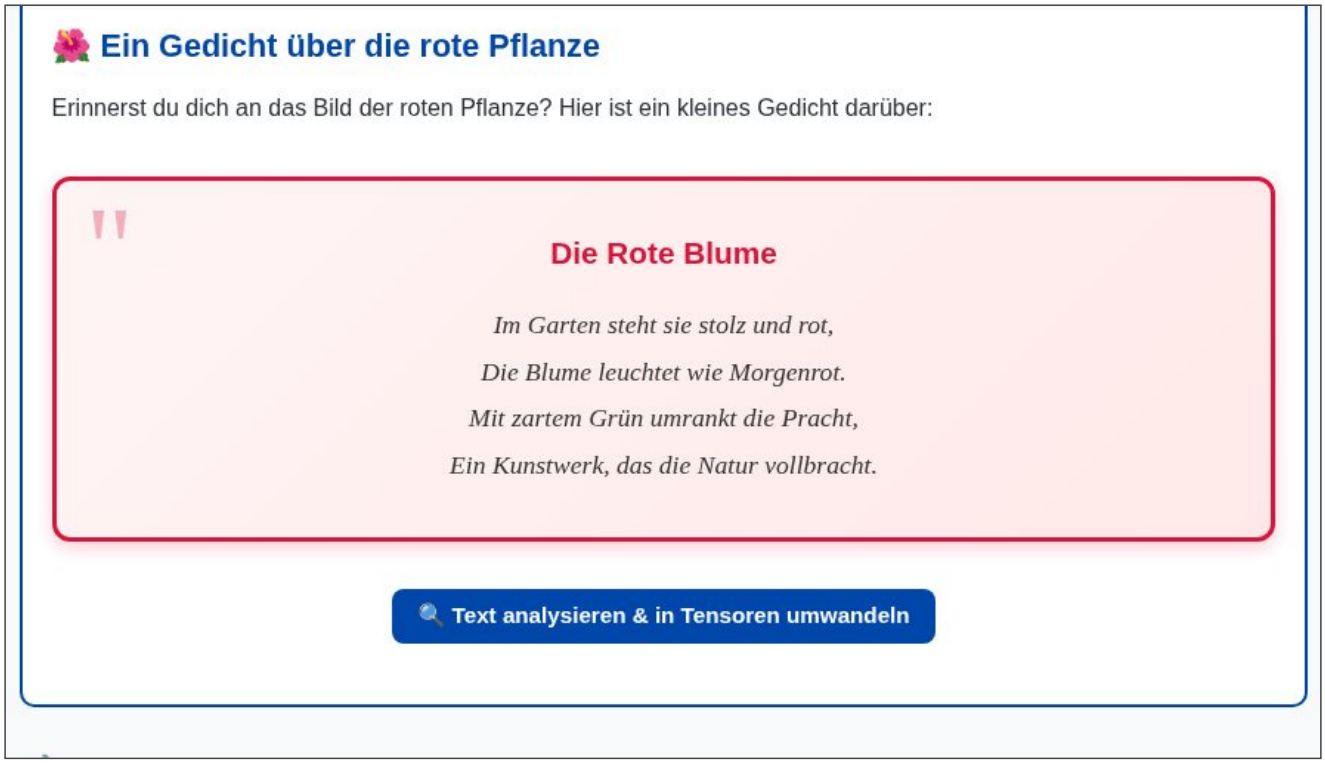
Erinnerst du dich an das Bild der roten Pflanze? Hier ist ein kleines Gedicht darüber:

Die Rote Blume

Im Garten steht sie stolz und rot,
Die Blume leuchtet wie Morgenrot.
Mit zartem Grün umrankt die Pracht,
Ein Kunstwerk, das die Natur vollbracht.

Am Gerät ausprobieren: herrki.de/040_text_und_llms#abschnitt-2

Direkt zum Modul



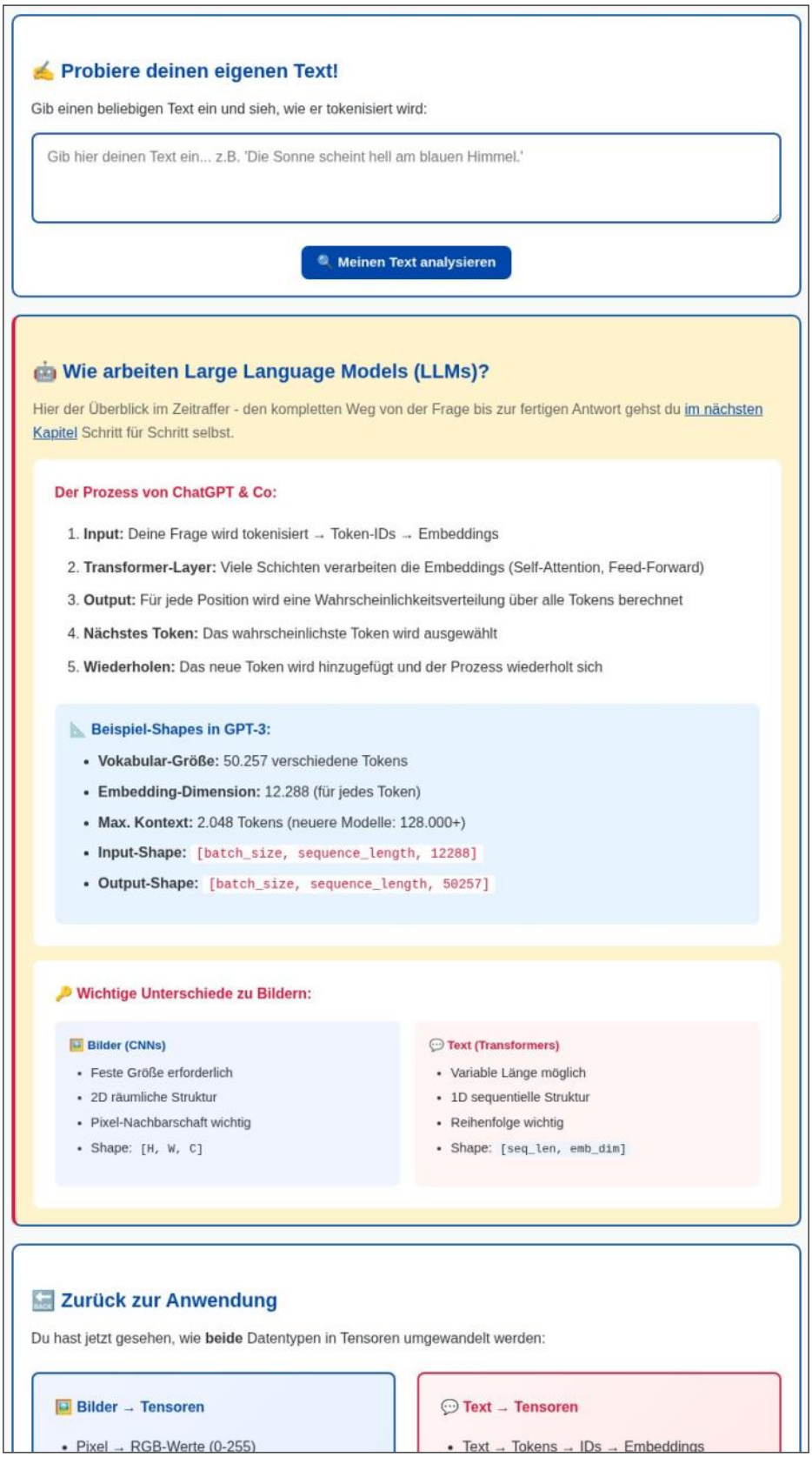
Interaktives Modul – am Gerät ausprobieren

Runter zum Tensor: Vom Text zum Zahlenpaket

Jetzt gehen wir unter die Haube. Drei Schritte trennen dein Gedicht von dem Zahlenpaket, mit dem ein LLM tatsächlich rechnet: zerlegen, nummerieren, in Bedeutungs-Vektoren übersetzen. Klick oben auf „Text analysieren“ — dann klappen die Schritte hier auf und du siehst sie an deinem eigenen Gedicht durchlaufen.

Am Gerät ausprobieren: herrki.de/040_text_und_llms#abschnitt-3

Direkt zum Modul



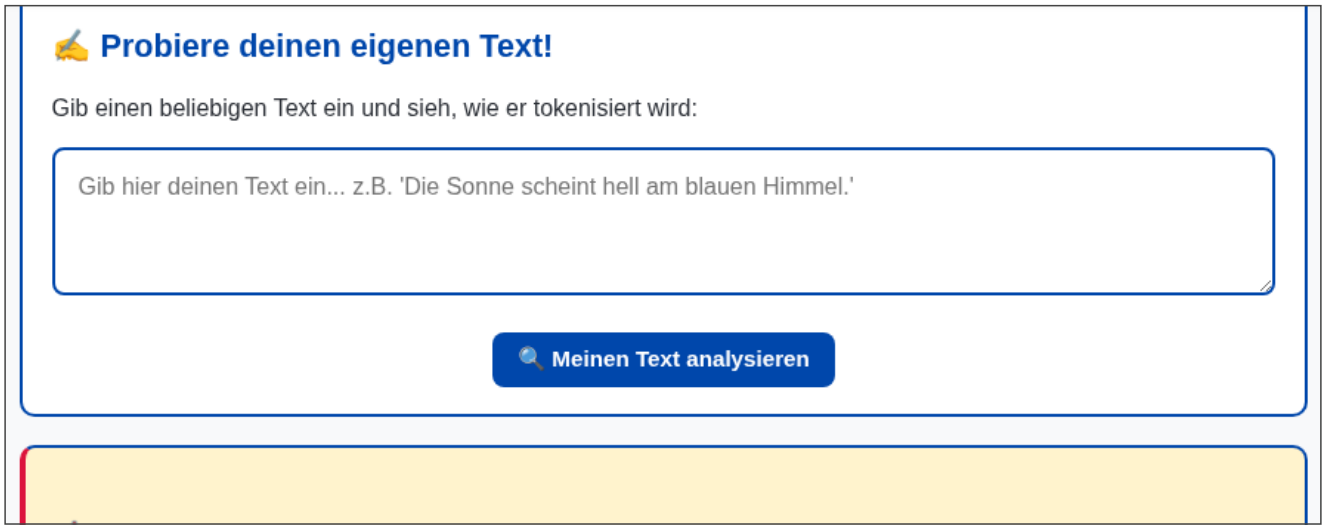
Interaktives Modul – am Gerät ausprobieren

Probiere deinen eigenen Text!

Gib einen beliebigen Text ein und sieh, wie er tokenisiert wird:

Am Gerät ausprobieren: herrki.de/040_text_und_llms#abschnitt-4

Direkt zum Modul



Interaktives Modul – am Gerät ausprobieren

Dein Text als Tokens:

Anzahl Tokens: 0 **Tensor Shape:** [0] **Als Embeddings:** [0, 768]

Wie arbeiten Large Language Models (LLMs)?

Hier der Überblick im Zeitraffer - den kompletten Weg von der Frage bis zur fertigen Antwort gehst du im nächsten Kapitel Schritt für Schritt selbst.

Der Prozess von ChatGPT & Co:

1. **Input:** Deine Frage wird tokenisiert → Token-IDs → Embeddings
2. **Transformer-Layer:** Viele Schichten verarbeiten die Embeddings (Self-Attention, Feed-Forward)
3. **Output:** Für jede Position wird eine Wahrscheinlichkeitsverteilung über alle Tokens berechnet
4. **Nächstes Token:** Das wahrscheinlichste Token wird ausgewählt
5. **Wiederholen:** Das neue Token wird hinzugefügt und der Prozess wiederholt sich

Beispiel-Shapes in GPT-3:

- **Vokabular-Größe:** 50.257 verschiedene Tokens
- **Embedding-Dimension:** 12.288 (für jedes Token)
- **Max. Kontext:** 2.048 Tokens (neuere Modelle: 128.000+)
- **Input-Shape:** `[batch_size, sequence_length, 12288]`
- **Output-Shape:** `[batch_size, sequence_length, 50257]`

Wichtige Unterschiede zu Bildern:

Bilder (CNNs)

- Feste Größe erforderlich
- 2D räumliche Struktur
- Pixel-Nachbarschaft wichtig
- Shape: `[H, W, C]`

Text (Transformers)

- Variable Länge möglich
- 1D sequentielle Struktur
- Reihenfolge wichtig
- Shape: `[seq_len, emb_dim]`

Am Gerät ausprobieren: herrki.de/040_text_und_llms#abschnitt-5

Direkt zum Modul

🤖 Wie arbeiten Large Language Models (LLMs)?

Hier der Überblick im Zeitraffer - den kompletten Weg von der Frage bis zur fertigen Antwort gehst du im nächsten Kapitel Schritt für Schritt selbst.

Der Prozess von ChatGPT & Co:

1. **Input:** Deine Frage wird tokenisiert → Token-IDs → Embeddings
2. **Transformer-Layer:** Viele Schichten verarbeiten die Embeddings (Self-Attention, Feed-Forward)
3. **Output:** Für jede Position wird eine Wahrscheinlichkeitsverteilung über alle Tokens berechnet
4. **Nächstes Token:** Das wahrscheinlichste Token wird ausgewählt
5. **Wiederholen:** Das neue Token wird hinzugefügt und der Prozess wiederholt sich

📐 Beispiel-Shapes in GPT-3:

- **Vokabular-Größe:** 50.257 verschiedene Tokens
- **Embedding-Dimension:** 12.288 (für jedes Token)
- **Max. Kontext:** 2.048 Tokens (neuere Modelle: 128.000+)
- **Input-Shape:** `[batch_size, sequence_length, 12288]`
- **Output-Shape:** `[batch_size, sequence_length, 50257]`

🔑 Wichtige Unterschiede zu Bildern:

🖼️ Bilder (CNNs)

- Feste Größe erforderlich
- 2D räumliche Struktur
- Pixel-Nachbarschaft wichtig
- Shape: `[H, W, C]`

💬 Text (Transformers)

- Variable Länge möglich
- 1D sequentielle Struktur
- Reihenfolge wichtig
- Shape: `[seq_len, emb_dim]`

Interaktives Modul – am Gerät ausprobieren

Zurück zur Anwendung

Du hast jetzt gesehen, wie **beide** Datentypen in Tensoren umgewandelt werden:

Bilder → Tensoren

- Pixel → RGB-Werte (0-255)
- Shape: `[H, W, 3]`
- Direkt als Zahlen
- Feste Größe nötig

Text → Tensoren

- Text → Tokens → IDs → Embeddings
- Shape: `[seq_len, emb_dim]`
- Bedeutung encodiert
- Variable Länge möglich

Jetzt bist du bereit, **neuronale Netze** zu verstehen! Du weißt, wie Daten (Bilder und Text) in Tensoren umgewandelt werden - die Sprache der KI. Später lernst du, wie diese Tensoren durch Neuronen, Layer und Aktivierungsfunktionen verarbeitet werden.

Und damit zurück zum Anfang: Das Gedicht über die rote Blume ist für ein LLM kein Text mehr, sondern ein Tensor der Shape `[Tokens, 768]` — eine Zeile pro Token, in jeder Zeile hunderte Zahlen, die die Bedeutung als Ort auf der Landkarte festhalten. „Leuchtet“ und „strahlt“ wohnen dort Nachbartür an Nachbartür, obwohl kein einziger Buchstabe gleich ist. Wenn du das nächste Mal etwas in ein Chatfenster tippst, weißt du, was in der ersten Millisekunde passiert: zerlegen, nummerieren, in Vektoren übersetzen.

Was danach kommt, siehst du direkt im nächsten Kapitel: „Wie LLMs Fragen beantworten“ — dort gehst du denselben Weg weiter, von der Frage bis zum letzten Wort der Antwort.

Zum Weiterlesen

- **Woher die Bedeutungs-Landkarte kommt:** Tomas Mikolov u. a., *„Efficient Estimation of Word Representations in Vector Space“* (word2vec), 2013 —

arxiv.org/abs/1301.3781 — das Paper, aus dem die berühmte Rechnung „König − Mann + Frau“ stammt.

- **Dasselbe zum Anschauen:** Jay Alammar, *„The Illustrated Word2vec“* — jalammar.github.io/illustrated-word2vec — weil er die Landkarte Schritt für Schritt zeichnet, statt sie nur zu beschreiben.
- **Warum Tokens keine Wörter sind:** Rico Sennrich u. a., *„Neural Machine Translation of Rare Words with Subword Units“* (Byte-Pair Encoding), 2015 — arxiv.org/abs/1508.07909 — hier steht, warum ein langes Wort in mehrere Stücke zerfällt.
- **Tokens selbst nachzählen:** OpenAI, *Tokenizer* — platform.openai.com/tokenizer — tipp einen deutschen Satz ein und sieh sofort, in welche Stücke ihn ein echtes LLM zerlegt (ohne Anmeldung nutzbar).
- **Auf HerrKI:** 010 — Was sind Tensoren? (das Zahlenpaket) und 310 — Latent Space (die Landkarte, auf der du selbst spazieren gehst).

Kurzer Check – mit Lösung

Lösungen für Lehrkräfte – für Lernende die Online-Version nutzen.

Drei schnelle Fragen, dann bist du durch — klick einfach deine Antwort an:

Frage 1: In welcher Reihenfolge wird Text für eine KI verarbeitet?

- **✔ Text → Tokens → Token-IDs → Embeddings**
- Text → Embeddings → Tokens → IDs
- Text → Rechtschreibprüfung → Übersetzung → Antwort

Genau die Kette aus dem Kapitel: zerlegen (Tokens), nummerieren (IDs), dann jedem Token seinen Bedeutungs-Vektor geben (Embedding). Erst damit kann gerechnet werden.

Frage 2: Was ist ein Word Embedding?

- Eine komprimierte ZIP-Version des Wortes
- **✔ Ein Zahlen-Vektor, der die Bedeutung als Position auf einer „Landkarte“ kodiert — ähnliche Wörter liegen nah beieinander**
- Die Häufigkeit des Wortes im Text

Richtig — und diese Landkarte heißt Latent Space. „König“ und „Königin“ wohnen dort Tür an Tür; im Latent-Space-Kapitel darfst du später selbst darauf spazieren.

Frage 3: „Liest“ die KI deinen Text so wie du?

- Ja, Wort für Wort wie ein Mensch
- **✔ Nein — sie rechnet mit Zahlen-Tensoren (z. B. Shape [Tokens, 768]); Bedeutung steckt in Vektor-Nachbarschaften**
- Nur wenn der Text auf Englisch ist

Genau: Nach der Umwandlung existiert dein Satz für das Modell nur noch als Tensor. Alles Weitere — Verstehen, Antworten — ist Rechnen mit diesen Vektoren.

Das nimmst du mit

- **Computer lesen nicht, sie rechnen.** Jeder Text wird erst zu Zahlen — sonst passiert gar nichts.
- **Die Kette ist immer dieselbe:** Text → Tokens → Token-IDs → Embeddings → Tensor.
- **Ein Token ist kein Wort.** Es ist ein häufiges Textstück — kurze Wörter bleiben ganz, lange zerfallen in Teile.
- **Die Token-ID ist nur eine Hausnummer.** Bedeutung steckt erst im Embedding: ein Vektor mit hunderten Zahlen, also ein Ort auf einer Landkarte.
- **Nähe = Ähnlichkeit.** Weil „Katze“ und „Hund“ in ähnlichen Sätzen vorkommen, landen sie dicht beieinander — daher funktioniert „König − Mann + Frau = Königin“ als Rechnung.

Halte kurz inne - das war ein großer Schritt: Du weißt jetzt, was beim Tippen in ChatGPT als Allererstes passiert: Text → Tokens → IDs → Bedeutungs-Vektoren. Wenn dir also jemand erzählt, die KI „liest" deine Nachricht - du weißt es besser: Sie rechnet mit einem Tensor der Shape [Tokens, 768]. Und wie aus diesen Vektoren dann eine ANTWORT entsteht? Genau das nehmen wir uns jetzt vor: „Wie LLMs Fragen beantworten".

So zitierst du dieses Kapitel

HerrKI: Wie Computer mit Sprache umgehen (LLMs). herrki.de/040_text_und_llms, ein privates Hobbyprojekt von Terragon.de, abgerufen am 09.09.2026.