# Deine eigene Bild-KI (Transfer Learning)

Kapitel online

Leg zwei bis vier eigene Klassen an, zeig der Kamera ein paar Beispiele — und dein Browser erkennt sie sofort. Das Prinzip hinter Googles Teachable Machine: ein eingefrorenes Riesennetz als Auge, ein winziges Gedächtnis als Lernteil.

**Interaktiv · Kapitel 48 von 50 · ca. 15 Min. · 300 XP**

**Einfach gesagt:** Du zeigst der Kamera ein paar Beispiele eines Gegenstands, und dein Browser erkennt ihn danach sofort wieder — er baut auf dem auf, was ein fertiges Modell schon sieht.

2019 stellte Google die **„Teachable Machine“** ins Netz: eine Seite, auf der man der Kamera ein paar Gegenstände zeigt, drei Knöpfe drückt — und die Maschine sie danach zuverlässig auseinanderhält. Millionen Menschen haben das ausprobiert, Lehrerinnen, Kinder, Museen. Es fühlte sich an wie Zauberei: **zehn Bilder, und sie kann es.**

Und genau da wird es interessant. In „Ziffern erkennen“ hast du ein Netz von Grund auf trainiert: **rund 7.000 Beispielbilder**, mehrere Trainingsrunden, Kaffeepause. Heute reichen **zehn** — und es geht sofort. Wie kann das sein? Hat jemand geschummelt?

Nein. Es ist ein Trick, der die halbe moderne KI trägt und einen Namen hat: **Transfer Learning**. Wir sehen ihn uns heute erst genau an — und dann baust du dir deine eigene Bild-KI.

**Roter Faden:** Erst das Geheimnis (runter bis auf die Zahlen im Vektor), dann das Werkzeug — und am Ende siehst du die 1.280 Zahlen, aus denen dein Browser gerade schließt: „Das ist die Kaffeetasse.“

**Was du hier lernst**

- Was ein **Embedding** ist — der „Bedeutungs-Fingerabdruck“ eines Bildes
- Warum ein **eingefrorenes** Netz mehr wert ist als ein frisch trainiertes
- Wieso **10 Beispiele** genügen, wo vorher 7.000 nötig waren
- Wie ein **Nächste-Nachbarn-Gedächtnis** (KNN) funktioniert — in drei Sätzen erklärt
- Im Mitmach-Teil: deine eigenen Klassen, live erkannt — plus ein Blick in den Fingerabdruck

**Für den Unterricht**

**Einstieg (3 Min):** Kapitel gemeinsam lesen, Alltagsbeispiel aufgreifen.

**Erkunden am Gerät (9 Min):** die interaktiven Module auf herrki.de/460_eigene_bild_ki ausprobieren.

**Sichern mit Quiz (3 Min):** „Kurzer Check" besprechen, „Das nimmst du mit" festhalten.

# Runter zum Tensor: Das Geheimnis — das große Netz lernt heute gar nichts

Der Kern des Tricks klingt zunächst paradox: Das große neuronale Netz, das gleich in deinen Browser geladen wird (**MobileNet V2**, 14 MB), wird **nicht trainiert**. Kein einziges seiner Gewichte ändert sich, während du Beispiele sammelst. Es ist **eingefroren**.

### Das Netz ist nur das Auge

MobileNet hat vor Jahren auf **rund 1,3 Millionen Fotos** (dem ImageNet-Datensatz) gelernt, Bilder zu *durchschauen*: Kanten, Texturen, Rundungen, Fell, Glanz, Schrift, typische Formen. Dieses Sehvermögen ist völlig unabhängig davon, ob es am Ende „Zebra“ oder „Kaffeetasse“ sagen soll. Wir benutzen deshalb nur den vorderen Teil — das **Auge** — und schneiden den hinteren Teil ab, der die 1.000 ImageNet-Wörter ausspuckt.

Was das Auge liefert, ist kein Bild und kein Wort, sondern eine **Liste von 1.280 Zahlen**. Diese Liste heißt **Embedding** — der Bedeutungs-Fingerabdruck deines Bildes. Zwei Fotos derselben Kaffeetasse aus zwei Winkeln ergeben zwei *ähnliche* Zahlenlisten. Ein Foto deiner Brille ergibt eine ganz andere. Genau das ist der Latent Space aus Kapitel 310, nur diesmal für Bilder statt für Wörter: **Ähnliches liegt nah beieinander.**

### Dein „Training“ ist ein Gedächtnis, kein Training

Und jetzt kommt der zweite, fast schon freche Teil. Wenn ähnliche Bilder ohnehin schon nahe beieinander liegen — warum dann noch ein Netz trainieren? Es reicht, sich die Fingerabdrücke deiner Beispiele **einfach zu merken**. Kommt ein neues Bild, berechnen wir seinen Fingerabdruck und schauen nach: *Welche gemerkten Vektoren liegen am nächsten?* Die Mehrheit dieser Nachbarn bestimmt die Antwort.

Dieses Verfahren heißt **KNN** („k nächste Nachbarn“) und ist so alt wie schlicht: kein Gradient Descent, keine Epochen, keine Lernrate. Nur Speichern und Abstand messen. **Deshalb** ist dein Training hier sofort fertig — es ist gar keines.

### Die Reise eines Bildes — Shape für Shape

```
// Was mit einem Kamerabild passiert, sobald du "Sammeln" drückst:

  Kamerabild [224, 224, 3] // Höhe x Breite x RGB, Werte 0…1
        ↓
  MobileNet V2 EINGEFROREN — kein Gewicht ändert sich
        ↓
  Embedding [1280] // der Bedeutungs-Fingerabdruck
        ↓
  Gedächtnis (KNN) // Abstand zu allen gemerkten Vektoren
        ↓
  "Kaffeetasse" 87 % // Mehrheit der k nächsten Nachbarn
```

**Nur der letzte Pfeil lernt.** Alles davor ist fertiges, geschenktes Vorwissen — und genau *deshalb* genügen zehn Bilder statt siebentausend: Du bringst dem System nicht bei, was ein Rand oder eine Rundung ist. Das weiß es längst. Du sagst ihm nur noch, *wie du* die Dinge nennst.

**Genau dieses Muster kennst du schon:** In „Finetuning vs. Prompting“ bleibt das riesige Sprachmodell ebenfalls unangetastet, und nur ein kleiner Aufsatz lernt dazu (Stichwort LoRA). Hier ist es dasselbe in Grün: **großes Netz eingefroren, winziger Aufsatz lernt.** Das ist Transfer Learning — Wissen wird von einer Aufgabe auf eine andere übertragen.

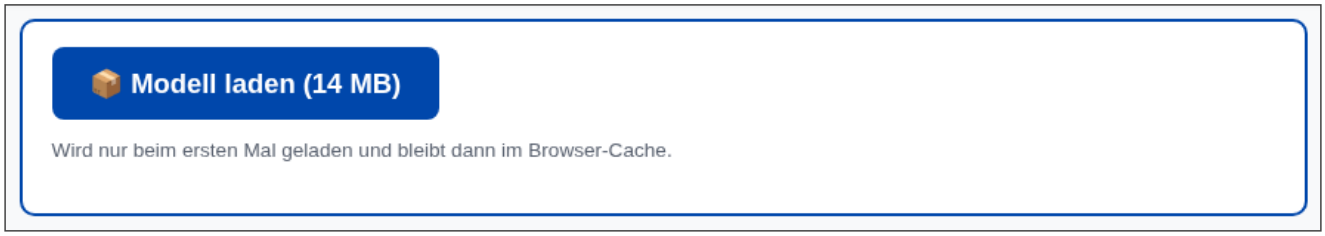
# Jetzt du: bring deinem Browser zwei Dinge bei

**So geht's:** Modell laden → Kamera starten → bei „Klasse 1“ den Sammeln-Knopf **gedrückt halten** und dabei z. B. deine Kaffeetasse hin und her drehen (ca. 5 Bilder pro Sekunde) → dasselbe bei „Klasse 2“ mit einem anderen Gegenstand. Sobald in **zwei** Klassen je mindestens ein Beispiel liegt, läuft die Erkennung live mit. Kein Kamerazugriff? Zieh stattdessen Fotos in die Ablagefläche der jeweiligen Klasse — funktioniert genauso.

Tipp für gute Ergebnisse: 15–30 Beispiele je Klasse, Gegenstand dabei drehen, näher/weiter halten, und ruhig mal den Hintergrund wechseln. Warum gerade der Hintergrund wichtig ist, steht weiter unten — das ist die klassische Falle.

Am Gerät ausprobieren: herrki.de/460_eigene_bild_ki#abschnitt-3

Direkt zum Modul




*Interaktives Modul – am Gerät ausprobieren*

**Deine Bühne**

Noch keine Quelle — starte die Kamera oder lade unten Bilder hoch.

Kein Zugriff auf die Kamera? Kein Problem — die Ablageflächen können alles genauso.

**Live-Erkennung**

–

Sammle Beispiele in mindestens zwei Klassen, dann geht es los.

### Deine Klassen

Mindestens 2, höchstens 4 Klassen. Namen kannst du direkt überschreiben.

**Der Blick ins Embedding**

Das hier ist der **„Bedeutungs-Fingerabdruck“** deines aktuellen Bildes — 1.280 Zahlen, hier die ersten 8. Halt einen Gegenstand vor die Kamera und beweg ihn: Die Zahlen zappeln ein wenig (anderer Winkel, anderes Licht) — aber sie springen erst dann wirklich um, wenn ein *anderer* Gegenstand ins Bild kommt. Genau diese Stabilität ist es, die den Nachbar-Vergleich funktionieren lässt.

```
[ … ] — noch kein Bild analysiert
```

Diese 1.280 Zahlen sind alles, was dein Gedächtnis von einem Bild jemals zu sehen bekommt.

# Ehrlich bleiben: zwei Dinge, die du wissen musst

### 1. Die Apfel-Falle: Dein Gedächtnis lernt womöglich den Hintergrund

Angenommen, du fotografierst alle „Apfel“-Beispiele auf einem roten Tischset und alle „Birne“-Beispiele auf einem weißen. Deine KI wird beides perfekt auseinanderhalten — nur eben womöglich an der **Farbe des Tisches**, nicht an der Frucht. Das Embedding enthält ja *alles*, was im Bild war, und das Gedächtnis nimmt jede Abkürzung, die zuverlässig funktioniert. Fachleute nennen das **Shortcut Learning**, und es ist der Klassiker verzerrter Trainingsdaten: Das Modell hat nicht gelernt, was du meintest, sondern was in deinen Daten am leichtesten zu erkennen war. **Gegenmittel:** Variiere beim Sammeln bewusst den Hintergrund, die Beleuchtung und den Winkel — dann bleibt als einzige Gemeinsamkeit deiner Beispiele der Gegenstand selbst übrig. Und teste ehrlich: Halt den Apfel mal über den weißen Untergrund.

### 2. Das Gedächtnis lebt nur in diesem Tab

Alles, was du hier sammelst, liegt im Arbeitsspeicher deines Browsers. **Lädst du die Seite neu, ist es weg** — wir speichern bewusst nichts, auch nicht lokal. Das ist keine Schlamperei, sondern die konsequente Version des Versprechens ganz oben: Deine Bilder verlassen dieses Fenster nie, nicht mal in Richtung Festplatte. Wenn du gleich fertig bist: erst spielen, dann neu laden.

# Zurück zur Anwendung

Am Anfang stand Googles Teachable Machine: ein paar Gegenstände in die Kamera halten, drei Knöpfe drücken, fertig — und es fühlte sich an wie Zauberei. Jetzt hast du genau das selbst gebaut und weißt, dass die Zauberei aus zwei nüchternen Teilen besteht: einem eingefrorenen Riesennetz als Auge und einem winzigen Gedächtnis für Vektoren als Lernteil. Du kannst ab jetzt für jede Sortier-Aufgabe abschätzen, ob ein paar Dutzend Beispiele reichen — und du weißt, worauf du beim Sammeln achten musst, damit dein Gedächtnis das Objekt lernt und nicht die Tischplatte dahinter.

Im nächsten Kapitel „Semantische Suche in DEINEN Texten“ bleibt der Trick derselbe, nur das Sinnesorgan wechselt: Aus Sätzen werden Vektoren, und aus dem Vergleich von Vektoren wird eine Suche, die Bedeutung versteht statt Buchstaben zu zählen.

**Zum Weiterlesen**


- Yosinski u. a. 2014, *„How transferable are features in deep neural networks?“* — arxiv.org/abs/1411.1792 — die Messung dazu, welche Schichten eines Netzes auf fremde Aufgaben passen und welche nicht.
- Howard u. a. 2017, *„MobileNets“* — arxiv.org/abs/1704.04861 — der Bauplan des eingefrorenen Auges, das hier in deinem Browser läuft.
- TensorFlow.js Tutorial, *Transfer learning image classifier* — tensorflow.org (tensorflow.org/js/tutorials/transfer/image_classification) — dieselbe Kombination aus MobileNet und KNN-Gedächtnis als Anleitung zum Nachbauen.
- Google, *Teachable Machine* — teachablemachine.withgoogle.com — das Original von 2019, das du hier gerade nachgebaut hast — inklusive Export für eigene Projekte.

# Kurzer Check – mit Lösung

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

**Frage 1: In Kapitel 420 brauchte das Ziffern-Netz rund 7.000 Bilder. Warum reichen hier zehn?**

- Weil Kaffeetassen einfacher aussehen als handgeschriebene Ziffern.
- **✓ Weil das Sehvermögen schon fertig geschenkt ist: Ein vortrainiertes Netz macht aus jedem Bild einen Bedeutungs-Vektor — und deine zehn Bilder werden nur noch als Nachbarn gemerkt und verglichen.**
- Weil KNN ein deutlich effizienterer Trainingsalgorithmus ist als Gradient Descent.

*Genau: Der aufwendige Teil — Kanten, Texturen, Formen sehen lernen — ist längst passiert, auf 1,3 Millionen Fotos. Deine Beispiele müssen nur noch Namen an Punkte im Latent Space heften. Deshalb ist es sofort fertig.*

**Frage 2: Was wird beim „Trainieren“ auf dieser Seite NICHT verändert?**

- **✓ Die Gewichte von MobileNet — das Netz ist eingefroren und lernt kein einziges Gewicht dazu.**
- Die gespeicherten Embeddings deiner Beispiele.
- Die Zuordnung „dieser Vektor gehört zu Klasse 2“.

*Das große Netz bleibt Bit für Bit so, wie es geladen wurde. Gelernt wird ausschließlich im kleinen Gedächtnis davor: Vektor merken, Etikett dranhängen. Dasselbe Prinzip wie beim Finetuning großer Sprachmodelle (Kapitel 370) — großes Netz stehen lassen, kleinen Aufsatz anpassen.*

**Frage 3: Alle deine „Apfel“-Fotos entstehen vor einer roten Wand, alle „Birne“-Fotos vor einer weißen. Die Erkennung klappt perfekt. Was ist das Problem?**

- Kein Problem — perfekte Erkennung ist perfekte Erkennung.
- Rot und Weiß liegen im Latent Space zu dicht beieinander.
- **✓ Womöglich unterscheidet die KI die Wände, nicht das Obst — sie hat die bequemste Abkürzung in deinen Daten gelernt (Shortcut Learning).**

*Der Fingerabdruck enthält das ganze Bild, Hintergrund inklusive. Wenn der Hintergrund perfekt mit dem Etikett korreliert, ist er das einfachste Erkennungsmerkmal — und genau das merkt sich das Gedächtnis. Gegenmittel: Hintergrund, Licht und Winkel bewusst durchmischen.*

# Das nimmst du mit

- **Ein Embedding ist der Bedeutungs-Fingerabdruck eines Bildes.** Aus deinem Kamerabild werden 1.280 Zahlen — und genau daraus schließt dein Browser auf die Klasse.
- **Das große Netz lernt hier gar nichts.** Es ist eingefroren und dient nur als Auge; seine Gewichte rührt niemand an.
- **Dein „Training“ ist ein Gedächtnis, kein Training.** Das KNN merkt sich die Vektoren deiner Beispiele und vergleicht neue damit — deshalb ist es sofort fertig.
- **Zehn Beispiele genügen, wo vorher Tausende nötig waren.** Das Vorwissen steckt schon im eingefrorenen Netz; du lieferst nur noch die Etiketten.
- **Vorsicht vor der Apfel-Falle.** Wenn alle Beispiele einer Klasse denselben Hintergrund haben, lernt dein Gedächtnis den Hintergrund — und beim Neuladen ist ohnehin alles weg.

**Und damit zurück nach oben:** Du hast gerade genau das gebaut, was Googles Teachable Machine berühmt gemacht hat — und weißt jetzt, dass die „Zauberei“ aus zwei nüchternen Teilen besteht: einem **eingefrorenen Auge** und einem **Gedächtnis für Vektoren**. Der Kontrast zu „Ziffern erkennen“ zeigt dir, was Vorwissen wert ist; „Latent Space“ erklärt, warum Ähnliches nah beieinander liegt; und „Finetuning vs. Prompting“ ist dieselbe Idee eine Etage höher, bei den Sprachmodellen. Als Nächstes kommt in dieser Reihe die Textsuche — dasselbe Embedding-Prinzip, nur mit Sätzen statt Bildern: Sätze werden zu Vektoren, und die Suche wird zum Nachbarschaftsvergleich. Der Trick bleibt derselbe.

### So zitierst du dieses Kapitel

HerrKI: Deine eigene Bild-KI (Transfer Learning). herrki.de/460_eigene_bild_ki, ein privates Hobbyprojekt von Terragon.de, abgerufen am 09.09.2026.