Wo lasse ich lokale Modelle laufen?

Notebook oder Server im Rechenzentrum — zwei Modelle, vier eigene Messreihen

Nachtrag zur Untersuchung „KI-Coding-Modelle im Eigenbetrieb“ - https://www.koeltzsch.com/blog/ki-modellvergleich-lokal-gegen-cloud

Kristian Költzsch · August 2026 - https://www.koeltzsch.com/blog/ki-modelle-hardware-notebook-oder-server

Ausgangslage

Der erste Foliensatz beantwortete die Frage nach der Modellqualität: Sieben .NET- und T-SQL-Aufgaben, sechs Cloud-Modelle gegen vier lokale, objektiv verifiziert per Compile-, Unit- und Runtime-Tests. Ergebnis: Lokale Modelle erreichen inzwischen die Einstiegsklasse der Cloud — für Analyse und Review brauchbar, für Generierung mit subtilen Korrektheitsanforderungen nicht.

> **Offen blieb die Hardwarefrage**
>
> Alle Messungen liefen auf einem MacBook Pro und einem Windows-Desktop. Was leistet dieselbe Aufgabe auf professioneller Rechenzentrums-Hardware? Für diesen Nachtrag stand ein Server mit einer RTX 6000 Pro Blackwell zur Verfügung — und erstmals lief ein identisches Modell in identischer Quantisierung auf beiden Maschinen.

Neu gemessen für diesen Nachtrag:

Zwei Modelle, jeweils auf beiden Maschinen — Qwen2.5-Coder-32B als dichtes Modell und Qwen3.6-35B-A3B als MoE-Modell. Gleiche Aufgaben, gleiches Sampling, jeweils vergleichbare Quantisierung. Damit misst der Vergleich die Maschine und die Architektur, nicht zwei verschiedene Modelle.

Warum der Modelltyp die Hardware bestimmt

Nicht jedes Modell belastet eine Maschine gleich — entscheidend ist, wie viele Parameter je Token gerechnet werden

Modell	Architektur	aktiv je Token	Speicher	tok/s
Qwen3-Coder-Next 80B	MoE	~2,5 Mrd.	65,6 GB	76,8
gpt-oss-120b	MoE	~5 Mrd.	63,4 GB	37,4
Qwen2.5-Coder-32B	**dense**	**32 Mrd.**	**19,9 GB**	**24 → 8**

Weniger Speicher heißt nicht schneller.

Das dichte 32-Milliarden-Modell belegt weniger als ein Drittel des Speichers und ist trotzdem das langsamste — weil es bei jedem Token alle Parameter durchrechnet statt nur einen Bruchteil. MoE-Modelle aktivieren je Token 2,5 bis 5 Milliarden und bleiben deshalb flott.

Was das für die Hardwarewahl bedeutet

Für den folgenden Vergleich wurde bewusst ein dichtes Modell gewählt: Es fordert die Maschine am stärksten und macht Unterschiede sichtbar, die bei einem MoE-Modell verborgen blieben. Wer ausschließlich MoE-Modelle einsetzt, wird die Grenzen des Notebooks später erreichen als hier gezeigt.

Umgebung 1 · MacBook Pro M5 Max

Chip / RAM	Apple M5 Max · 128 GB Unified Memory (~546 GB/s)
OS / Runtime	macOS 26.6 · LM Studio 0.4.19+2 · llama.cpp Metal 2.27.1
Modelle	Qwen2.5-Coder-32B (Q4_K_M, 19,9 GB) und Qwen3.6-35B-A3B (Q4_K_M, 22,1 GB)
Konfiguration	GPU max · Kontext 65.536 · 4 Slots
Speicherbild	rund 20 GB belegt · Ladezeit 3 s
Besonderheit	Mobil einsetzbar, keine laufenden Kosten

Gemessenes Tempo

24

tok/s · dichtes Modell

77

tok/s · MoE-Modell

Ein Speicherpool für CPU und GPU. Das dichte Modell bricht unter Dauerlast auf 8 tok/s ein, das MoE-Modell bleibt stabil.

Umgebung 2 · Server mit RTX 6000 Pro

Chip / RAM	RTX 6000 Pro Blackwell · 96 GB GDDR7 (~1,79 TB/s)
OS / Runtime	Linux · vLLM hinter LiteLLM-Proxy
Modelle	Qwen2.5-Coder-32B (AWQ 4-bit) und Qwen3.6-35B-A3B (FP8)
Konfiguration	Modell vollständig im VRAM, kein Auslagern
Speicherbild	rund 20 GB von 96 GB belegt
Besonderheit	Deutsches Rechenzentrum · rund 1.500 € im Monat

Gemessenes Tempo

65

tok/s · dichtes Modell

177

tok/s · MoE-Modell

Rund dreimal so viel Speicherbandbreite wie das Notebook. Beide Modelle halten ihr Tempo über alle Läufe, ohne Einbruch.

Die Qualität ändert sich nicht

Zwei Modelle, je drei Läufe pro Aufgabe, beide Maschinen — vier unabhängige Messreihen

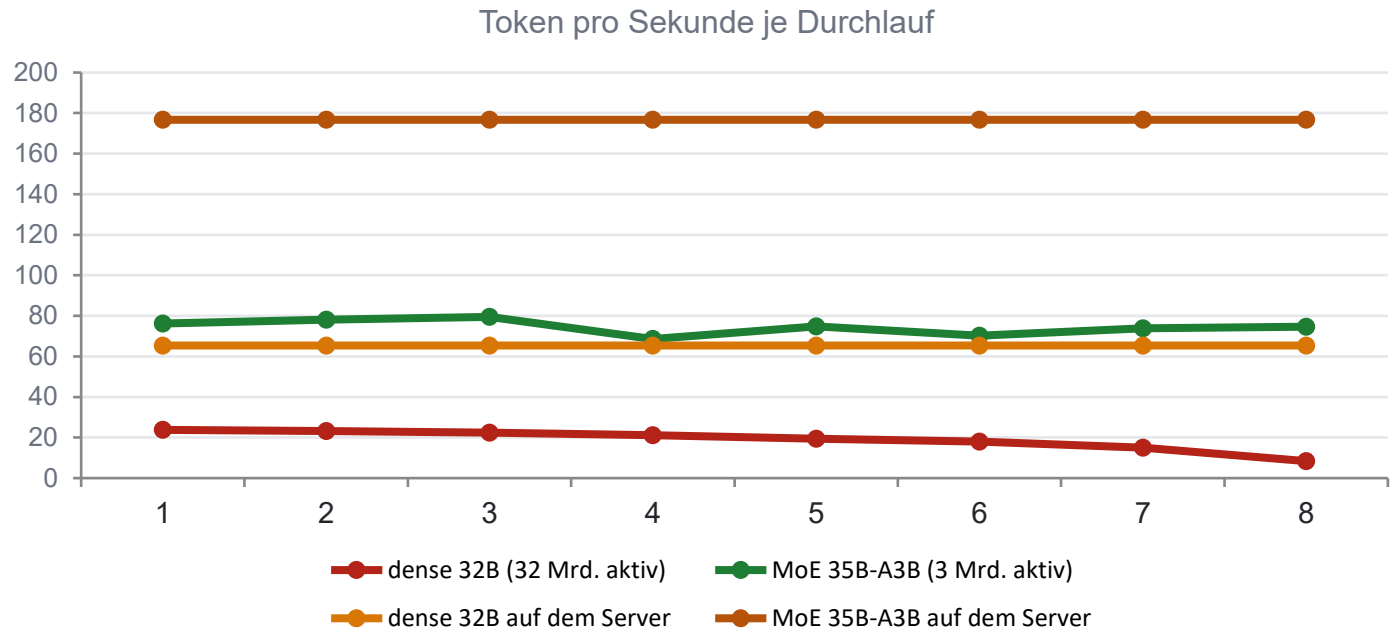
Aufgabe	dense: Mac	dense: Server	MoE: Mac	MoE: Server
A · Nebenläufigkeit	0 von 3	1 von 3	0 von 3	0 von 3
B · T-SQL-Aggregation	0 von 3	0 von 3	1 von 3	1 von 3
C · Code-Review	3 von 3	3 von 3	3 von 3	3 von 3
D · Blazor-Dialog	0 von 3	2 von 3	0 von 3	2 von 3
E · EF-Core-Query	0 von 3	0 von 3	0 von 3	0 von 3
F · Regex-Parser	0 von 3	0 von 3	0 von 3	0 von 3
G · Refactoring	3 von 3	3 von 3	2 von 3	1 von 3
GESAMT	**6 von 21**	**9 von 21**	**6 von 21**	**7 von 21**

Bessere Hardware macht ein Modell schneller, nicht klüger.

Beim dichten Modell 6 gegen 9, beim MoE-Modell 6 gegen 7 — beide Differenzen liegen im Streubereich dreier Läufe. C gelingt in allen vier Reihen, E und F scheitern in allen vier am selben Fehler.

Warum die Architektur über Dauerlast entscheidet

Beide Architekturen auf beiden Maschinen — vier Messreihen im direkten Vergleich



−65 %

dichtes Modell: von 23,8 auf 8,4 — der Einbruch geht immer weiter

stabil

MoE auf dem Notebook: 76,2 zu Beginn, 74,7 nach acht Läufen. Auf dem Server halten beide Modelle ihr Tempo über alle Läufe — 65,4 und 176,7 tok/s.

Ein MoE-Modell rechnet je Token nur einen Bruchteil der Parameter — und hält deshalb durch.

Bei anhaltender Volllast über mehrere tausend Token gibt auch das MoE-Modell nach, findet aber ein Plateau bei rund 80 Prozent seines Ausgangswerts. Das dichte Modell rutscht dagegen ohne Boden durch — nach acht Läufen bei einem Drittel.

Wie genau ist diese Messung?

Derselbe Test dreimal wiederholt — dasselbe Modell, dieselbe Maschine, dieselben Aufgaben

Aufgabe	Durchgang 1	Durchgang 2	Durchgang 3	Abweichung
A · Nebenläufigkeit	0 von 3	0 von 3	0 von 3	*keine*
B · T-SQL-Aggregation	1 von 3	0 von 3	1 von 3	*1 Punkt*
C · Code-Review	3 von 3	3 von 3	3 von 3	*keine*
D · Blazor-Dialog	0 von 3	0 von 3	0 von 3	*keine*
E · EF-Core-Query	0 von 3	0 von 3	0 von 3	*keine*
F · Regex-Parser	0 von 3	0 von 3	0 von 3	*keine*
G · Refactoring	2 von 3	2 von 3	1 von 3	*1 Punkt*
GESAMT	**6 von 21**	**5 von 21**	**5 von 21**	**±1 Punkt**

Fünf von sieben Aufgaben identisch

A, C, D, E und F liefern in allen drei Durchgängen exakt dasselbe Ergebnis. Nur B und G schwanken um je einen Punkt — beides Aufgaben, bei denen das Modell knapp an der Lösung liegt.

Was das für alle Zahlen hier bedeutet

Unterschiede von einem Punkt sind Rauschen, keine Aussage. 6 gegen 7 heißt: gleichwertig. Erst ab etwa drei Punkten Abstand lohnt sich eine Interpretation — etwa 5 gegen 10 beim Modellvergleich.

Was kostet die Rechenleistung?

Öffentlich verfügbare Angaben, Stand August 2026 — die Preise bewegen sich derzeit stark

Variante	Kosten	Anmerkung
Server mit RTX 6000 Pro (96 GB), deutscher Anbieter	**rund 1.200 bis 1.600 € im Monat**	*je nach RAM-Ausstattung, zzgl. Einrichtung*
Größere Variante mit 768 GB RAM	rund 2.100 € im Monat	*nach der Preisanpassung im April 2026*
Einführungspreis des Angebots	889 € im Monat	*Dezember 2025 — inzwischen überholt*
Karte als Neukauf	rund 8.500 → 13.250 USD	*Markteinführung 2025 gegen heute, plus 55 %*
MacBook Pro M5 Max, 128 GB	einmalige Anschaffung	*keine laufenden Kosten, mobil einsetzbar*

Die Preise bewegen sich nach oben

Beim Hosting stiegen die Preise im April 2026 um knapp die Hälfte, der Kaufpreis der Karte legte binnen anderthalb Jahren um 55 Prozent zu — Ursache ist die Knappheit bei GDDR7-Speicher. Wer heute rechnet, sollte die Angaben vor einer Entscheidung direkt beim Anbieter prüfen; die öffentlich auffindbaren Zahlen widersprechen sich teils, weil ältere Vergleichsseiten noch den Einführungspreis führen.

Alle Angaben aus öffentlich zugänglichen Quellen. Kein Anbieter genannt — die Auswahl ist austauschbar, vergleichbare Angebote gibt es bei mehreren deutschen Rechenzentren.

Welches Modell wofür?

Die Messwerte übersetzt in Einsatzgebiete — beide Modelle haben unterschiedliche Stärken

Einsatzgebiet	Spezialisiertes Coder-Modell	Generalistisches MoE-Modell
Code-Review, Bug-Suche, Erklärungen	**geeignet (3 von 3)**	**geeignet (3 von 3)**
Legacy-Refactoring	**gut geeignet (3 von 3)**	**eingeschränkt (1–2 von 3)**
SQL-Abfragen, einfache Aggregation	**eingeschränkt**	**eingeschränkt (1 von 3)**
Nebenläufigkeit, Framework-Randbereiche	**nicht geeignet**	**nicht geeignet**
Agentische Schleifen, Stapelverarbeitung	**nicht geeignet (drosselt)**	**geeignet (bleibt stabil)**
Arbeiten mit vertraulichem Code	**geeignet**	**geeignet**

Lokale Modelle sind Analysewerkzeuge, keine Codegeneratoren.

Beim Code-Review erreichen beide Modelle die volle Punktzahl — dort ersetzen sie einen Cloud-Dienst ohne Abstriche. Bei Generierung mit subtilen Korrektheitsanforderungen scheitern beide zuverlässig an denselben Aufgaben.

Bewertung aus je drei Läufen pro Aufgabe auf beiden Maschinen. „Nicht geeignet“ heißt: in keinem Lauf ein verwendbares Ergebnis.

Wann welche Maschine?

MacBook Pro M5 Max

für den Arbeitsplatz

- Einzelne Anfragen, interaktives Arbeiten
- Reviews und Analyse sensibler Codebestände
- Mobil, ohne laufende Kosten
- Für Dauerlast MoE-Modelle wählen — dichte brechen ein

Server im Rechenzentrum

für den Dauerbetrieb

- Zwei- bis dreifaches Tempo, unabhängig von der Architektur
- 96 GB VRAM — auch große Modelle passen ganz hinein
- Kein Einbruch unter Dauerlast, mehrere Nutzer parallel
- Laufende Kosten ab rund 1.200 € im Monat

Die Frage ist nicht, welche Maschine besser ist — sondern wie oft man sie braucht.

Bei gelegentlicher Nutzung gewinnt das Notebook, weil es ohnehin vorhanden ist. Bei durchgehender Auslastung gewinnt der Server, weil er das Tempo hält. Die Modellqualität spielt bei dieser Entscheidung keine Rolle — sie ist auf beiden Maschinen gleich.