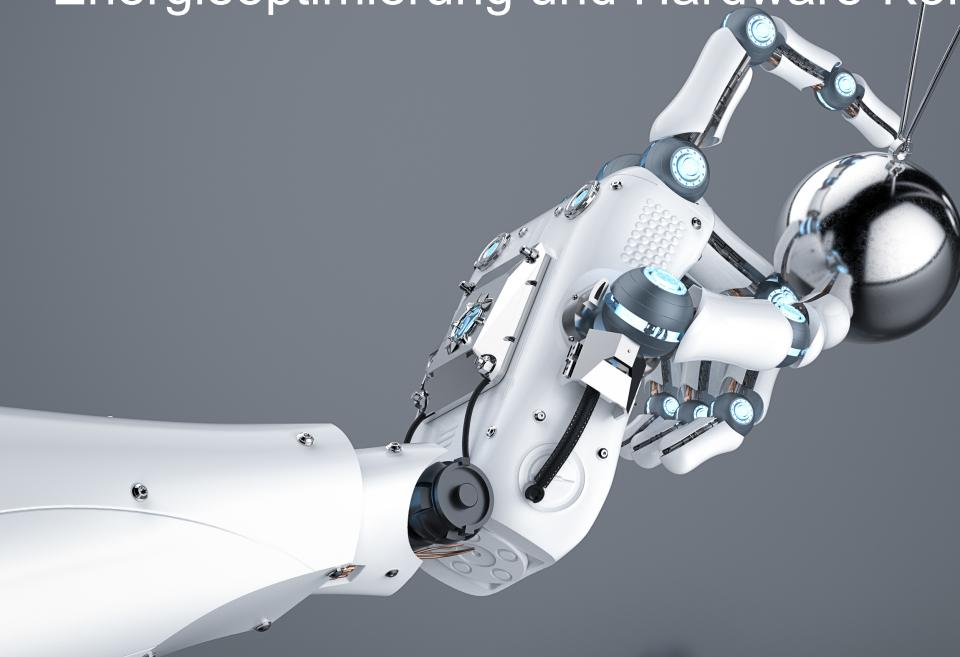


# Kolloquium Thomas Schmitt

28 November 2024

## Effiziente Reduktion großer Sprachmodelle: Methodenvergleich und Anwendungsstrategien für Energieoptimierung und Hardware-Kompatibilität

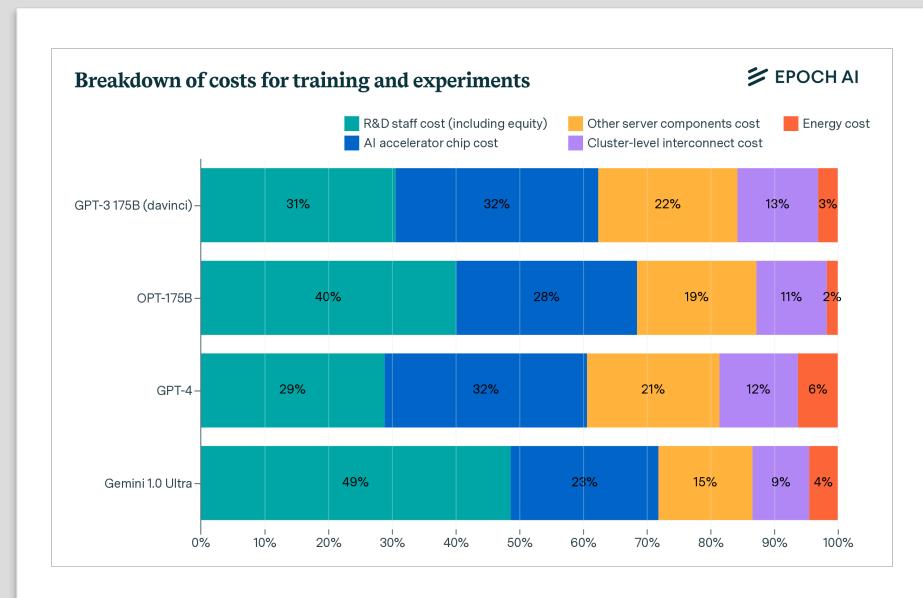


- Motivation
- Zielsetzung
- Verwendete Technologien
- Modelle und Messungen
- Matrix Experimente
- Resultate
- Fazit

# Motivation — Teurer Hardwarebedarf

- Nvidia Grafikkarten (GPUs) sind z.Zt. ein unverzichtbarer Bestandteil für die Entwicklung und Nutzung von Large Language Models (LLMs)
- Schon ein relativ kleines Modell, wie Llama 3 8B, benötigt eine GPU im Wert von ca. 15 k€ für die native Nutzung durch einen Benutzer (Im engl. als Inference bezeichnet)
- Lehrinstitute, kleine Firmen, sowie Weltregionen mit geringen finanziellen Möglichkeiten sind von Entwicklung und Nutzung eigener LLMs weitgehend ausgeschlossen.

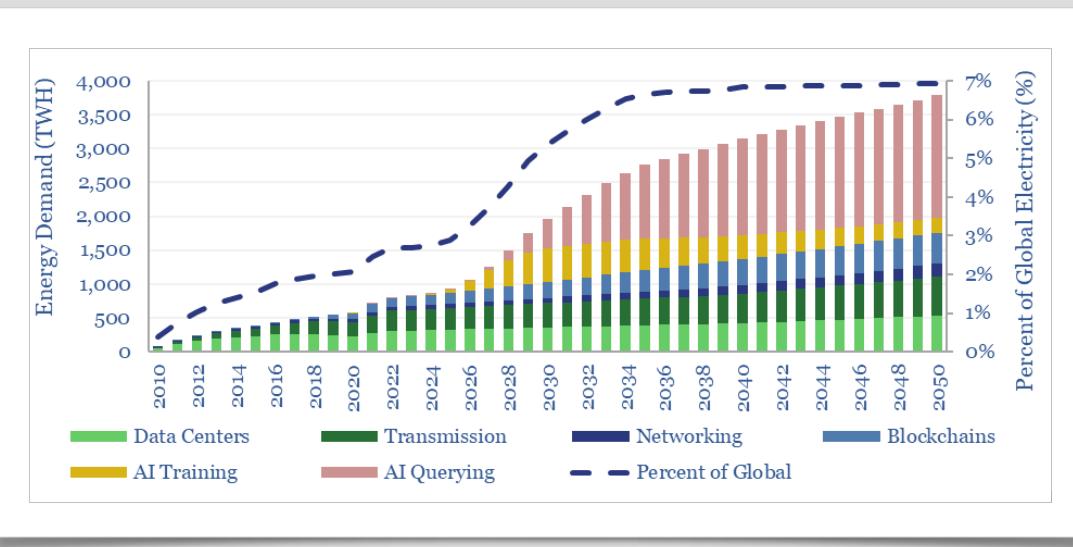
- Die nebenstehende Grafik <sup>2</sup> zeigt, dass bei aktuellen LLM Projekten ca. 50 - 70% der Gesamtkosten in Hardware fließen.
- Die Entwicklungskosten von GPT-4 beliefen sich auf geschätzte 78 Millionen US\$ <sup>1</sup>



<sup>1</sup> <https://www.inside-it.ch/so-viel-kosten-grosse-ki-modelle-20240422>

<sup>2</sup> Ben Cottier u.a. „The rising costs of training frontier AI models“. In: (2024). arXiv: 2405.21015 [cs.CY].

# Motivation — Hoher Energieverbrauch



- Die Grafik der „Thunder Said Energy research consultancy“ sagt für die nächsten 25 Jahre eine Verdopplung des Energiebedarfs von Rechenzentren voraus, welcher größtenteils durch das Training und die Nutzung von KI entsteht<sup>1</sup>
  - Im September 2024 kündigte Microsoft an ein altes AKW für die Stromversorgung seines KI Rechenzentrums zu reaktivieren<sup>2</sup>
  - Google kauft von „Kairos Power“ Mini-Atomkraftwerke, um den Strombedarf seiner Rechenzentren zu gewährleisten. Auch hier ist der Treiber des Energie Mehrverbrauchs im KI Ausbau begründet<sup>3</sup>

<sup>1</sup> Thunder Said Energy. *Internet energy consumption*. <https://thundersaidenergy.com/downloads/internet-energy-consumption-data-models-forecasts/>

<sup>2</sup> [https://www.heise.de/news/Atomstrom-für-KI-Rechenzentren-Microsoft-lässt-Three-Mile-Island-reaktivieren-9939236.html](https://www.heise.de/news/Atomstrom-fuer-KI-Rechenzentren-Microsoft-laesst-Three-Mile-Island-reactivieren-9939236.html)

<sup>3</sup> <https://blog.google/outreach-initiatives/sustainability/google-kairos-power-nuclear-energy-agreement/>

# Ziel der Masterarbeit

**Die Zielsetzung dieser Arbeit, war die Verringerung von Ressourcen Anforderungen bei Open Source LLMs, unter weitestgehender Beibehaltung ihrer generellen Fähigkeiten.**

Kriterien:

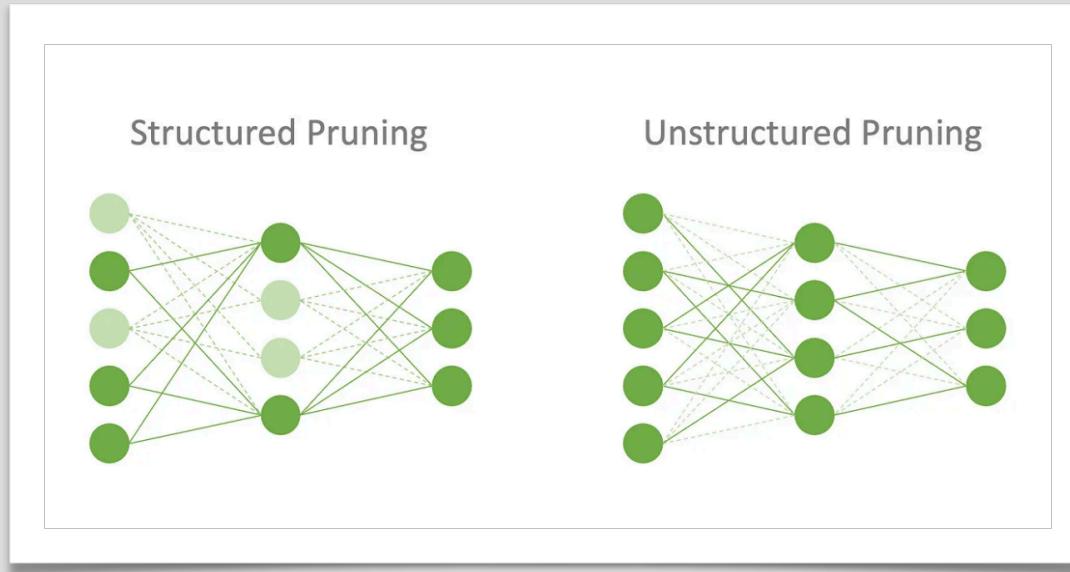
- Verkleinerung des (GPU) Speicherbedarfs
- Verbesserung der Inferenz-Geschwindigkeit
- Weitgehende Beibehaltung der Fähigkeiten dieser Modelle
- Nutzung der Modelle auf verschiedenen Plattformen (ggf. auch ohne GPU Unterstützung)

Um dies zu erreichen wurden folgende, in den letzten Jahren entwickelte Technologien bzw. Kombinationen dieser Technologien verwendet:

- Pruning
- Quantisierung
- Knowledge Distillation
- Low Rank Adaption (LoRA)

# Technologien — Pruning

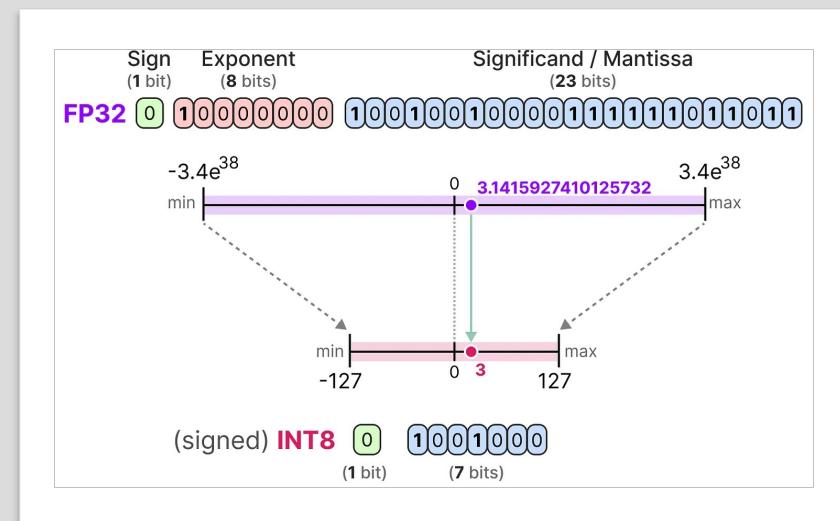
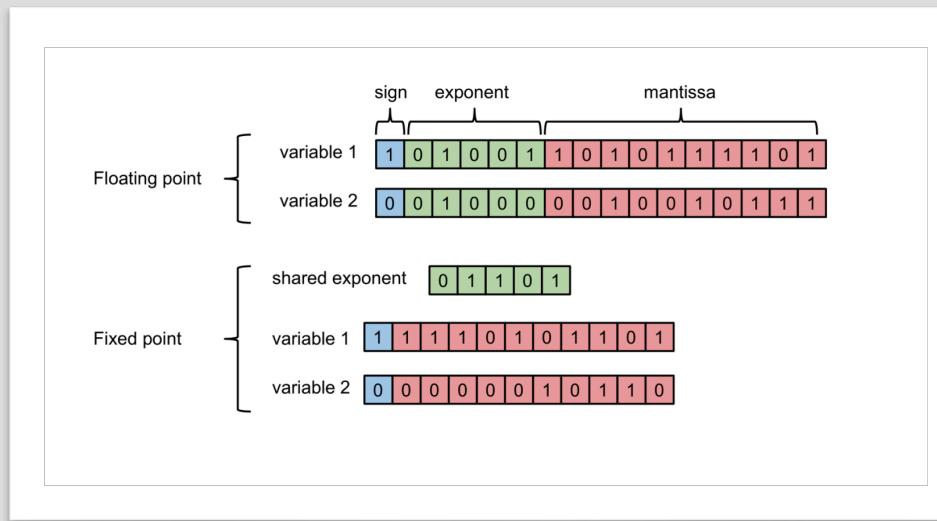
Der Begriff „Pruning“ umfasst verschiedene Verfahren, die zum Ziel haben, für die Qualität weniger relevante Anteile eines deep neural networks (DNN) zu entfernen.



Beim *Structured Pruning* werden ganze Bereiche des DNNs entfernt, während beim *Unstructured Pruning* nur einzelne Gewichte in den Schichten des Netzwerks entfernt werden. Bei beiden Ansätzen werden vorab die Strukturen/Gewichte identifiziert, welche nur wenig zur Gesamtleistung des DNNs beitragen.  
In der Arbeit wurde ein Verfahren ausgewählt, welches ein Hybrid aus beiden Ansätzen benutzt.

# Technologien — Quantisierung

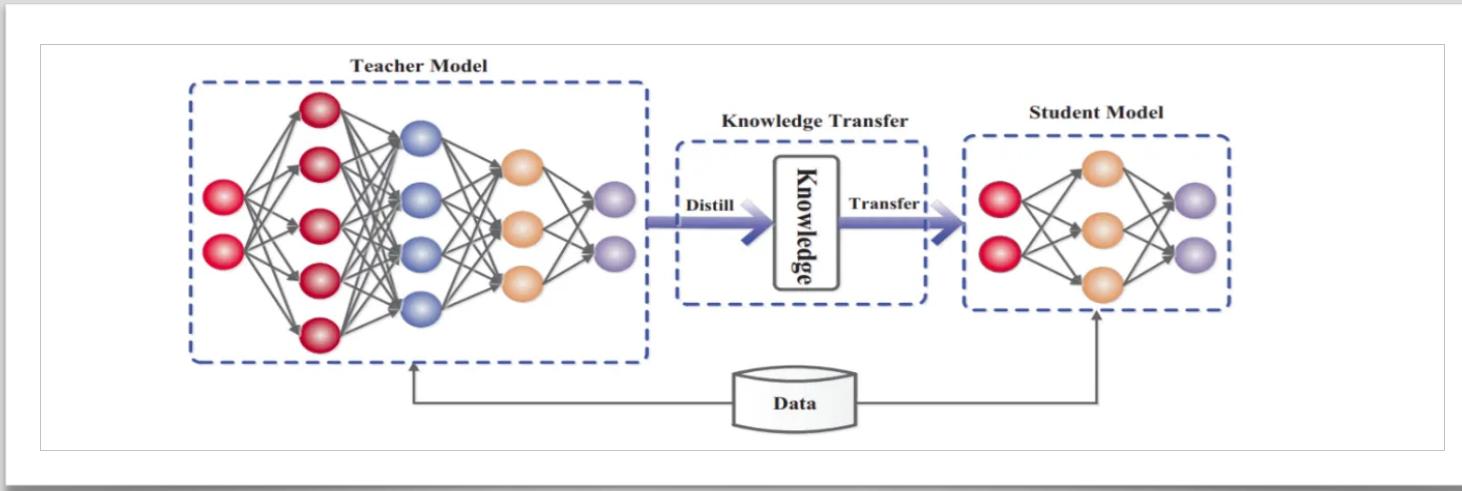
Als Quantisierung wird die Verringerung der Präzision der Einzelgewichte des Modells bezeichnet. Diese Technologie ist gut etabliert und es existieren mehrere alternative Ansätze Quantisierungen durchzuführen.



Alleine durch die Änderung des Variablentyps (linke Abbildung) von Fliesskomma zu Festkomma kann ein erheblicher Anteil an Speicherplatz eingespart werden. Weitere Einsparungen von Speicherplatz können durch Kalibrierung der Werte und eine Verringerung der Bitbreite (rechte Abbildung) der Variablen erfolgen.

# Technologien — Knowledge Distillation

Knowledge Distillation beschreibt den Prozess bei dem Antworten eines größeren Modells (Teacher) benutzt werden, um ein kleineres Modell (Student) zu trainieren.

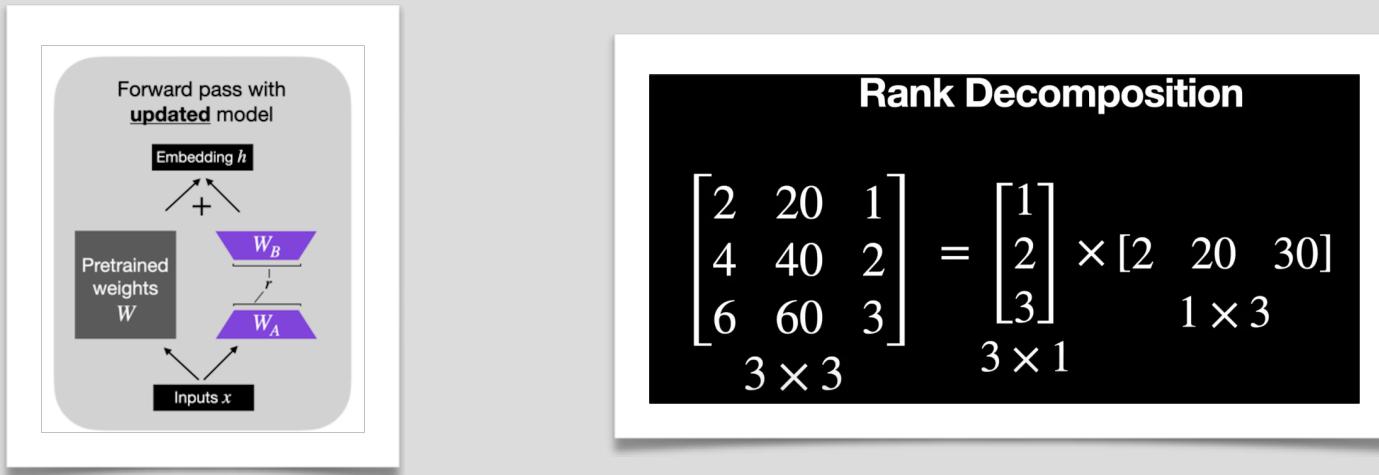


Die Übertragung des „Knowledge“ wird primär zwischen Modellen mit ähnlicher Struktur vorgenommen, kann aber auch zwischen Modellen mit unterschiedlicher Struktur erfolgen. In dieser Arbeit wurde Knowledge Distillation auf die durch Pruning verkleinerten und mittels LoRA retrainierten Modelle (Student) angewendet, wobei das zugehörige Ursprungsmodell als „Teacher“ fungierte.

Abbildung aus: <https://amit-s.medium.com/everything-you-need-to-know-about-knowledge-distillation-aka-teacher-student-model-d6ee10fe7276>

# Technologien — Low rank adaption (LoRA)

Das LoRA Verfahren selbst verringert nicht primär die Größe eines LLMs, sondern wird in dieser Arbeit zur Verbesserung geprunter Modelle mittels Retraining eingesetzt. In der verwendeten QLoRA Weiterentwicklung wird es mit einer Quantisierung kombiniert, so dass letztlich doch ein verkleinertes Modell entsteht.



LoRA fügt den „Weights“  $W_{org}$  eines Modells ein Adapter  $\Delta W$  hinzu der aus zwei niederrangigen Matrizen  $\Delta W = W_A \times W_B$  besteht. Die Matrizen  $W_A$  und  $W_B$  werden im Training angepasst und dann dem Modell hinzugefügt, dass so auf neue Inhalte angepasst wird. Dieses Verfahren benötigt deutlich weniger Speicherplatz und nimmt keine Änderungen am Ursprungsmodell vor.

# Modelle und Messungen

Die Experimente wurden mit öffentlich verfügbaren Large Language Models durchgeführt. Dabei wurde immer die kleinste verfügbare Variante der Modelle verwendet, damit die Hardwareanforderungen und die notwendigen Zeiträume handhabbar blieben:

## Verwendete Modelle:

- Meta Llama 2 7B - Llama Modell, das auf der Transformer Architektur basiert und längere Zeit als „Referenz“ galt.
- Meta Llama 3 8B Instruct - Dritte Evolutionsstufe der Llama Modelle, Veröffentlicht im April 2024
- Mistral 7B Instruct - Veröffentlicht im September 2023 von der gleichnamigen französischen Firma. Enthält neuere Technologien wie „Sliding window attention“ und „flash attention“

## Durchgeführte Messungen:

- Standard Benchmarks mit Hilfe des „lm\_evaluation\_harness“ Framework
- „Perplexity“ Messung mit dem WikiText-2 Datensatz
- Inferenz-Geschwindigkeit und GPU Speicherauslastung anhand von selbst gewählten Beispielen

# Experimentalmatrix

Die Tabelle gibt an welche Experimente in der Arbeit durchgeführt mit welchem Modell durchgeführt wurden. Im unteren Teil der Tabelle sind die Experimente angegeben, bei denen mehrere Verfahren kombiniert wurden.

Verfahren	Llama 3 8B Instruct	Mistral 7B Instruct v0.2	Llama 2 7B chat
Depth pruning (Layer Extraction mit PruneMe)	X	X	-
Activation-aware Weight Quantization (AWQ)	X	X	X
Generic Pretrained Transfomer Quantisation (GPTQ)	X	X	-
Kombinierte Verfahren			
Pruning + Lora (Shortened Llm)	X	X	X
Pruning + Quantization (Qlora)	X	X	X
Pruning + Lora + Knowledge distillation	X	X	-
Pruning + Lora + AWQ	X	X	X

# Resultate — Benchmarks

Die Tabelle zeigt die Benchmark Ergebnisse für das Modell Llama 3 8B Instruct und die verschiedenen aus dem Grundmodell generierten Varianten. Auffallend ist der starke Abfall der Resultate bei der geprunten Variante und die deutliche Verbesserung durch das Retraining.

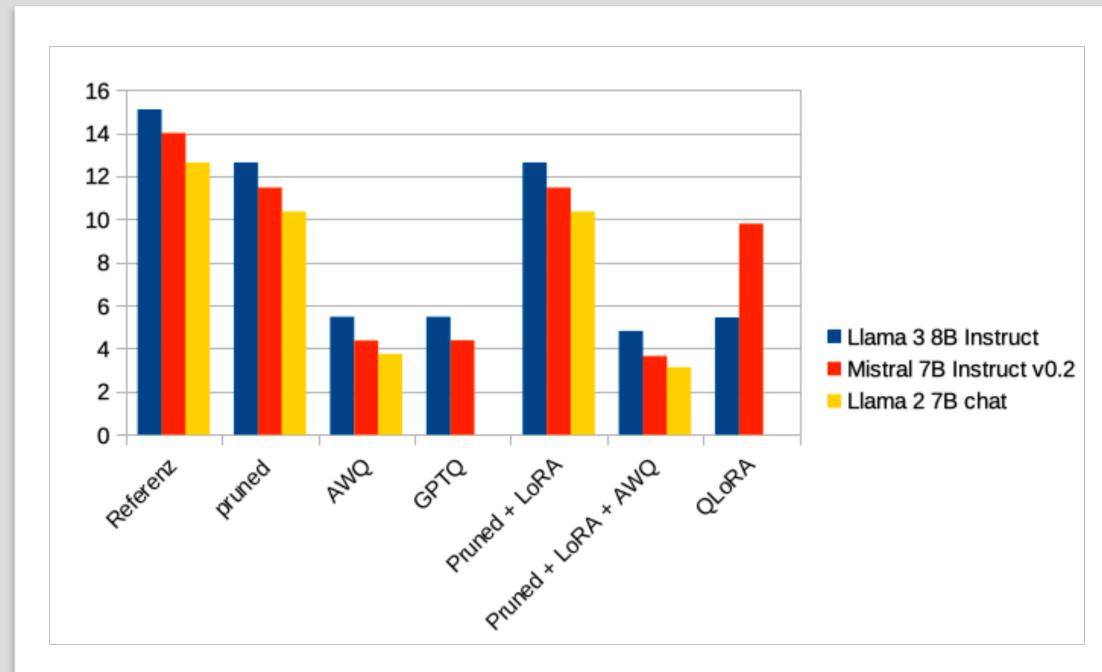
Llama 3 8B instruct	winogrande	truthfulqa_mc2	hellaswag	arc_challenge	Durchschnitt
Referenz	<b>0,7206</b> Standard Fehler 0,0126	<b>0,5165</b> 0,0152	<b>0,7582</b> 0,0043	<b>0,5674</b> 0,0145	<b>0,6407</b>
pruned (shortened-LLM)	<b>0,7111</b> Standard Fehler 0,0127	<b>0,5000</b> 0,0157	<b>0,6255</b> 0,0048	<b>0,4625</b> 0,0146	<b>0,5748</b>
AWQ	<b>0,7370</b> Standard Fehler 0,0124	<b>0,5099</b> 0,0152	<b>0,7529</b> 0,0043	<b>0,5580</b> 0,0145	<b>0,6395</b>
GPTQ	<b>0,7230</b> Standard Fehler 0,0126	<b>0,5196</b> 0,0152	<b>0,7388</b> 0,0044	<b>0,5341</b> 0,0146	<b>0,6289</b>
pruned + lora	<b>0,7316</b> Standard Fehler 0,0125	<b>0,5189</b> 0,0151	<b>0,7269</b> 0,0044	<b>0,4974</b> 0,0146	<b>0,6187</b>
pruned + lora + awq	<b>0,7230</b> Standard Fehler 0,0126	<b>0,5273</b> 0,0150	<b>0,7322</b> 0,0044	<b>0,4872</b> 0,0146	<b>0,6174</b>
pruned + lora + gptq	<b>0,7222</b> Standard Fehler 0,0126	<b>0,5002</b> 0,0151	<b>0,7158</b> 0,0045	<b>0,4940</b> 0,0146	<b>0,6080</b>
QLoRA	<b>0,7190</b> Standard Fehler 0,0126	<b>0,5169</b> 0,0153	<b>0,7539</b> 0,0043	<b>0,5478</b> 0,0145	<b>0,6344</b>

Die Tabelle zeigt die Perplexity Ergebnisse für dieselben Modelle wie oben. Hier ist die Verschlechterung des Ergebnisses für die geprunte Variante noch deutlicher erkennbar als in der oberen Tabelle.

Modell: Llama 3 8B instruct	bits_per_byte	byte_perplexity	word_perplexity
Referenz	0,62	1,54	9,94
AWQ	0,63	1,55	10,30
GPTQ	0,65	1,57	10,53
pruned	1,08	2,11	<b>53,86</b>
prune + lora	0,71	1,64	<b>13,96</b>
prune + lora + awq	0,70	1,62	<b>13,24</b>
QLoRA	0,64	1,56	10,19

# Resultate — Speicherauslastung und Inferenz

Die Tabelle zeigt die Speicherauslastung der GPU durch die verschiedenen Modelle. Man erkennt die deutliche Verringerung durch Quantisierung (AWQ, GPTQ) und die ebenfalls signifikante Verringerung durch das Pruning. Bei QLoRa ist auffällig, dass der ressourcensparende Effekt bei dem Llama Modell deutlich stärker ausgeprägt ist.



Inferenz Werte in Token pro Sekunde. Die obere Tabelle zeigt GPU mit Cuda 11.7. Die untere Tabelle zeigt GPU mit Cuda 12.5 und AWQ Unterstützung

Modell	Referenz	awq*	gptq	pruned	qlora	Pruned + lora	Pruned + lora +awq*
Llama-3-8B-Instruct	27,35	2,40	6,12	34,42	29,39	36,08	3,03

Modell Name	Referenz	prune+lora+awq
Llama 3 8B Instruct	10,22	33,34

# Resultate — Knowledge Distillation

Bei dem geprunten und retrainierten Modell (Student), wurde das Knowledge Distillation Verfahren als weiterer Schritt angewendet. Das Grundmodell diente dabei als „Teacher“.

Benchmarks:	winogrande	truthfulqa_mc2	hellawag	arc_challenge	Durchschnitt
LLama 3 8B pruned lora	0,7316	0,5189	0,7269	0,4974	0,6187
Fehler	<i>0,0125</i>	<i>0,0151</i>	<i>0,0044</i>	<i>0,0146</i>	
LLama 3 8B pruned lora dist	0,6535	0,4425	0,6645	0,4386	0,5498
Fehler	<i>0,0134</i>	<i>0,0166</i>	<i>0,0047</i>	<i>0,0145</i>	
Mistral pruned lora	0,7174	0,5774	0,7457	0,4855	0,6315
Fehler	<i>0,0127</i>	<i>0,0154</i>	<i>0,0043</i>	<i>0,0146</i>	
Mistral pruned lora dist	0,6330	0,4654	0,6694	0,4437	0,5529
Fehler	<i>0,0135</i>	<i>0,0162</i>	<i>0,0047</i>	<i>0,0145</i>	

Die Ergebnisse der Knowledge Distillation entsprachen in keiner Weise den Erwartungen, da die Werte nach diesem zusätzlichen Schritt schlechter als zuvor waren. Folgende Ursachen könnten zugrunde liegen:

- Es ist ein systematischer Fehler (Verwechslung o.ä.) unterlaufen — Durch eine vollständige Wiederholung des Experiments, konnte dieser Aspekt weitgehend ausgeschlossen werden.
- Die verwendete Implementation (DSKD) ist relativ neu und erhält ggf. selbst noch Fehler — Ein Email Kontakt mit dem Autor konnte dies nicht endgültig klären.
- Der „Wissensabstand“ zwischen dem „Student“ und „Teacher“ Modell ist zu gering um ein verbessertes Ergebnis zu erzielen.
- Das verwendete Datenset führt zu einem „overfitting“ auf die verwendeten Trainingsdaten.

# Resultate — Vergleich Quantisierung Verfahren

Die Tabelle zeigt die Resultate für die AWQ und GPTQ Quantisierung im Vergleich zueinander. Man erkennt in den Benchmark Ergebnissen leichte Vorteile für die AWQ Quantisierung.

Llama 3 8B instruct	winogrande	truthfulqa_mc2	hellawag	arc_challenge	Durchschnitt
referenz	0,7206 0,0126	0,5165 0,0152	0,7582 0,0043	0,5674 0,0145	0,6407
GPTQ	0,7230 0,0126	0,5196 0,0152	0,7388 0,0044	0,5341 0,0146	0,6289
AWQ	0,7370 0,0124	0,5099 0,0152	0,7529 0,0043	0,5580 0,0145	0,6395
pruned + lora + GPTQ	0,7222 0,0126	0,5002 0,0151	0,7158 0,0045	0,4940 0,0146	0,6080
pruned + lora + AWQ	0,7230 0,0126	0,5273 0,0150	0,7322 0,0044	0,4872 0,0146	0,6174

Die Belegung des GPU Speichers (In GB) ist für beide Modelle identisch (GPTQ mit 4 Bit)

Modell	Referenz	pruned	AWQ	GPTQ
Llama 3 8B Instruct	15,09	12,63	5,47	5,47

Bei den Perplexity Werten unterscheiden sich die Ergebnisse beider Quantisierungs-Verfahren nur minimal. Die Inference Geschwindigkeit wurde im direkten Vergleich nicht ermittelt.

# Fazit — Experimente

Eine Transformation von Large Language Models in Richtung geringerer Hardware Anforderungen ist mit Hilfe der beschriebenen Technologien möglich.

- Die neueren Quantisierungs Verfahren wie AWQ und GPTQ schaffen es die GPU Speicher Auslastung erheblich zu verringern. Dabei müssen nur geringe Kompromisse bei der Qualität der Ergebnisse in Kauf genommen werden. Die Inferenz Geschwindigkeit steigt, bei optimaler Unterstützung durch die Laufzeit Umgebung, ebenfalls signifikant an.
- Ein Pruning der Modelle ermöglicht ebenfalls eine Reduktion der GPU Speicher Belegung. Es muss aber zwingend ein Retraining des veränderten Modells stattfinden, da ansonsten die Qualität der Ergebnisse stark abfällt.
- Eine Kombination der Technologien führt zu besseren Ergebnissen als die Einzeltechnologien, bedarf aber aufwändiger Versuchsreihen und einer genau kontrollierten Balance zwischen Reduktion und Ergebnis.
- Tendenziell verhalten sich alle drei getesteten LLMs ähnlich in Bezug auf die Effekte durch die Reduktion. Um ein optimales Ergebnis zu erhalten sind allerdings entsprechende Versuchsreihen - Insbesondere beim Pruning - erforderlich.
- Das Potential von Knowledge Distillation konnte in dieser Arbeit, aus unklaren Gründen, nicht ausgeschöpft werden.

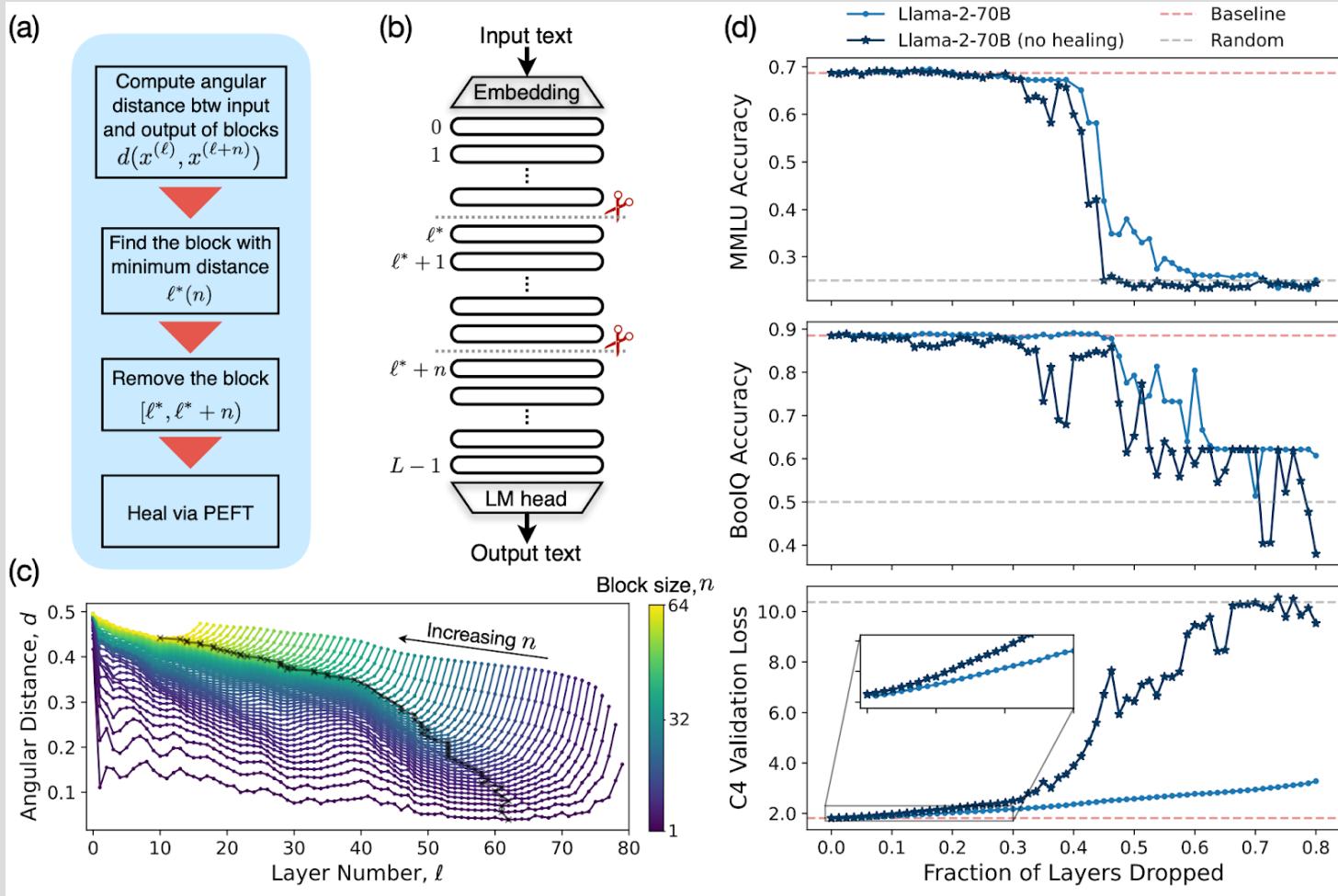
# Fazit — Einordnung und Ausblick

- Die hier generierten Modelle können grundsätzlich auch in offline Szenarien und auf CPUs bzw. GPUs aus dem Consumer Bereich eingesetzt werden. Je nach Leistungsfähigkeit der Hardware, kann die Inferenz Geschwindigkeit für stark interaktive Anwendungen unbefriedigend sein.
- Einen wichtigen Beitrag könnten solche Modelle in spezialisierteren Szenarien leisten. Dazu sollte im Prozess der Modellkomprimierung ein Retraining, mit den für das Szenario relevanten Daten erfolgen:
  - Für das Retrainig kann z.Bsp. der LoRA Schritt nach dem Pruning genutzt werden.
  - Eine andere Möglichkeit auch aktuelle, für den Anwendungszweck relevante Daten, zu berücksichtigen stellt Retrieval-Augmented Generation (RAG) dar.
- Die in dieser Arbeit erstellten Modelle können der Klasse der **Small Language Models** (SLMs) zugeordnet werden<sup>1</sup>. Dieser Begriff wird in der Literatur zunehmend verwendet und summiert Modelle, bei denen der Focus auf Ressourcenschonung, Spezialisierung und der Nutzung von Standard Hardware liegt.

<sup>1</sup> Überblick zu SLMs: <https://arxiv.org/pdf/2410.20011>

# Backup

# Technologien — PruneME



# Technologien — AWQ Quantisierung vs. GPTQ

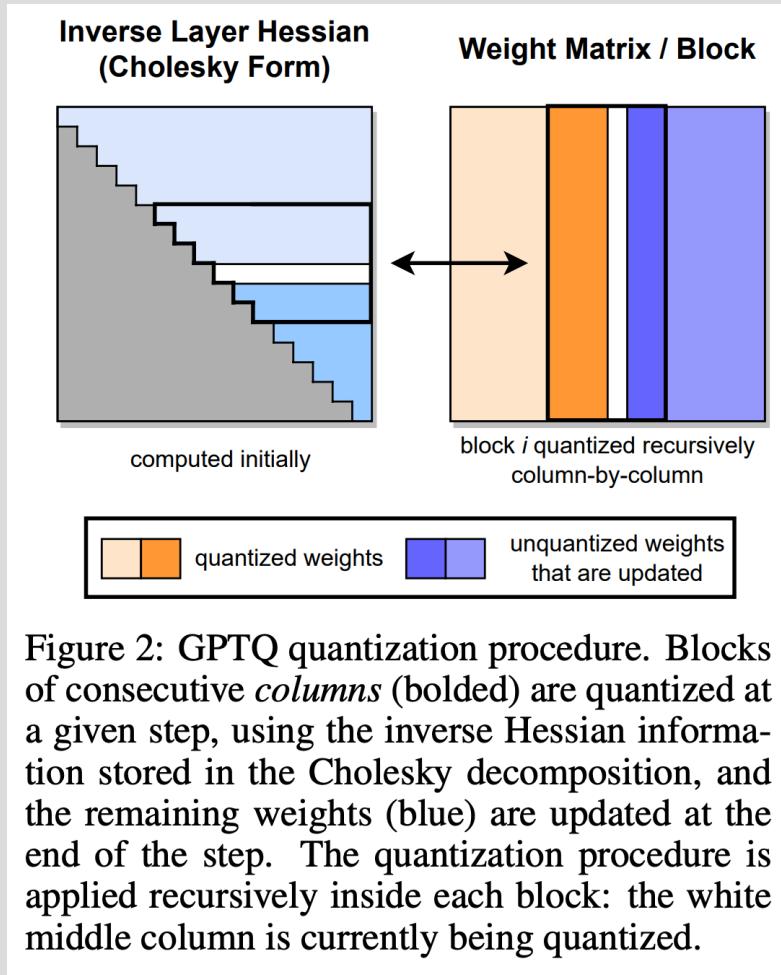


Figure 2: GPTQ quantization procedure. Blocks of consecutive *columns* (bolded) are quantized at a given step, using the inverse Hessian information stored in the Cholesky decomposition, and the remaining weights (blue) are updated at the end of the step. The quantization procedure is applied recursively inside each block: the white middle column is currently being quantized.

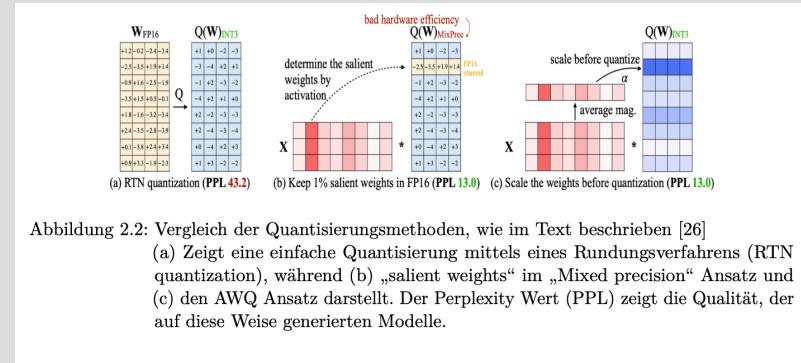


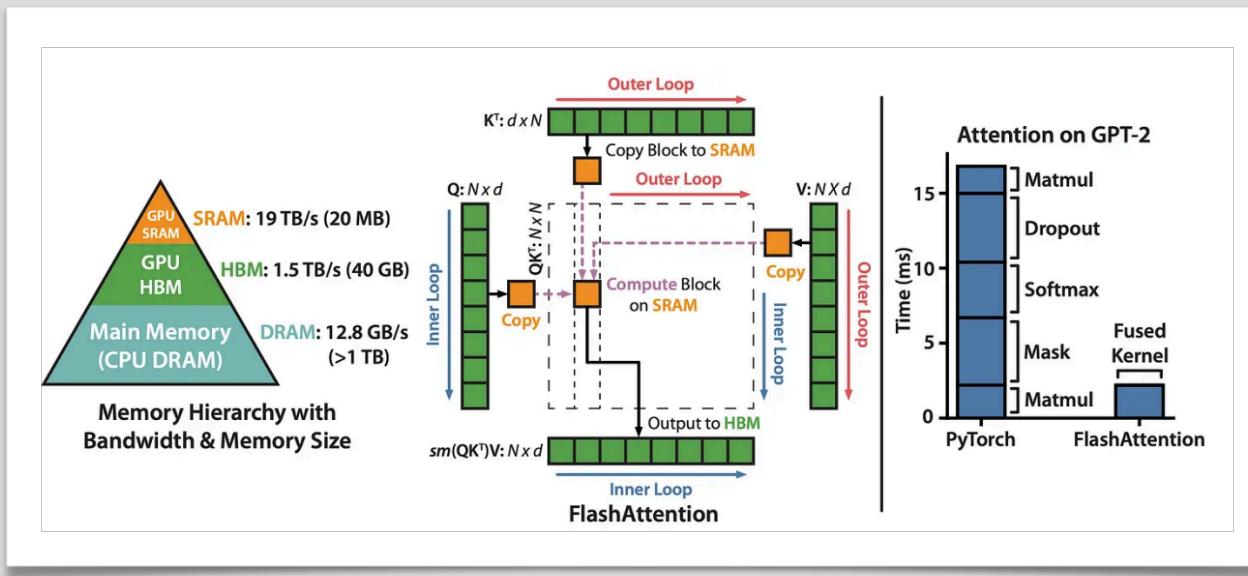
Abbildung 2.2: Vergleich der Quantisierungsmethoden, wie im Text beschrieben [26]

(a) Zeigt eine einfache Quantisierung mittels eines Rundungsverfahrens (RTN quantization), während (b) „salient weights“ im „Mixed precision“ Ansatz und (c) den AWQ Ansatz darstellt. Der Perplexity Wert (PPL) zeigt die Qualität, der auf diese Weise generierten Modelle.

GPTQ  
Schichtweise isolierte Quantisierung  
Versucht den Fehler im Output zu minimieren

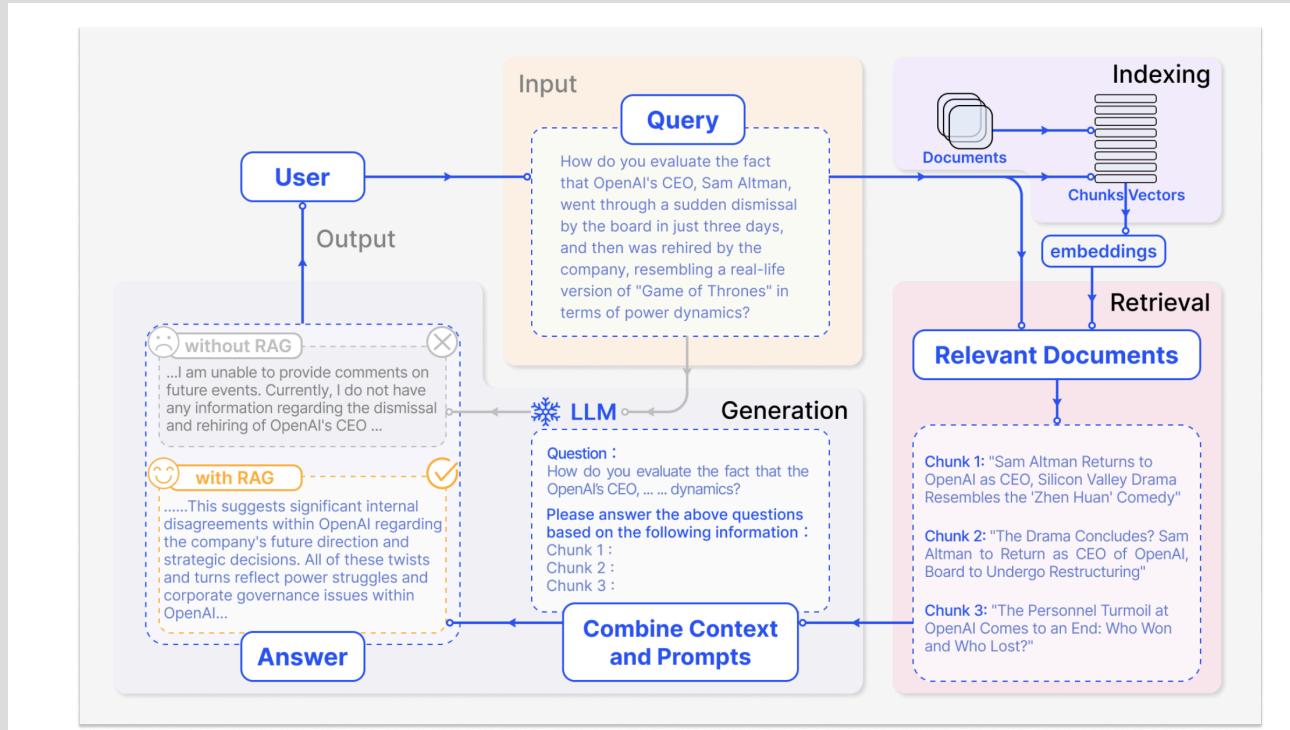
# Backup — Flash Attention

Flash Attention splittet die großen Attention-Matrizen in kleinere auf. Diese können dann durch die schnelleren Memory Bereiche (SRAM) des Systems abgearbeitet werden.



Bei der normalen Attention Mechanismus wächst der Speicherbedarf quadratisch zur Sequenz Länge. Bei Flash-Attention ist das Linear. Die Berechnungen können leichter parallelisiert werden.

# Backup — RAG



Die Assoziation dieser mit der Query erfolgt über die Nutzung von Embedding Ähnlichkeiten.

# Backup — Einordnung der Technologien

