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

Effizeint reduction of Lareg Language Models: Comparing methods and develop strategies to optimize energy footprint and hardware comatitibility

Masterarbeit

Dr. Thomas Schmitt
Fachhochschule Südwestfalen
1. August 2024

Autor:	Dr. Thomas Schmitt
Referent:	Prof. Dr. Heiner Giefers
Korreferent:	Max Kuhmichel
Eingereicht:	1. August 2024

Zusammenfassung

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Inhaltsverzeichnis

1 Exposee zur geplanten Masterarbeit

1.1 Einleitung - Überblick

Die geplante Arbeit untersucht aktuelle Verfahren zur Effizienzsteigerung von Large Language Models, vergleicht diese miteinander und versucht sie (ggf.) sinnvoll zu kombinieren. Effizienzsteigerung bedeutet in diesem Kontext, die Verringerung von Hardware Anforderungen, insbesondere an die Größe des Hauptspeichers und die Geschwindigkeit der GPU bzw. CPU. Ziel ist Verfahren zu entwickeln bzw. Möglichkeiten zu beschreiben, welche es erlauben, bei akzeptabler Qualität solche Modelle auf weniger performanten Hardware Infrastrukturen wie z.Bsp. embedded Computern lokal zu betreiben. Als kritisches Maß gilt hier, neben der Qualität der Inferenzergebnisse, vor allem die Latenzzeit bis das Modell eine Antwort liefert. Hierbei gelten für die jeweiligen Anwendungsfälle unterschiedlich akzeptable Zeiträume. Ein aktuelles LLM bietet allerdings auf einem System, welches nicht mit einer (NVIDIA) GPU ausgestattet ist, Latenzzeiten im Bereich von Minuten oder sogar Stunden. Dies ist in fast allen denkbaren Anwendungsfällen inakzeptabel.

1.2 Einleitung - Verfahren zur Steigerung der Effizienz

Es existieren verschiedene Verfahren, um LLMs in Größe und Komplexität zu reduzieren, die in der Einleitung der Arbeit genauer besprochen und untersucht werden sollen:

- Pruning - Umfasst verschiedene Verfahren zum ausdünnen von Neuronalen Netzwerken. Häufig werden 0 Werte und/oder redundante Werte entfernt. Innerhalb der Arbeit werden verschieden Varianten diskutiert und ausgewählt. [ZG17] [Sun+23] [FA23]
- Depth Pruning (shortened-llm)[Kim+24]

- Knowledge Distilliation - Hier wird der Ansatz eines Lehrer-Schüler Modells genutzt. Dabei wird das Schüler Modell auf die Vorhersagen des Lehrer Modells trainiert. Die zugrunde liegende Intention ist, dass es damit implizit das erworbene Wissen des Lehrer Modells übernimmt, welches selbst mit einer sehr großen Anzahl von Daten trainiert wurde [Xu+24]
- Low Rank Factorization (Lora) - Lora benutzt die Zerlegung von komplexeren Matrizen in niederrangigere Matrizen, [Xu+15] [Hu+21]. Mit diesem Verfahren kann ein Feintuning eines LLMs erfolgen, um dessen Effizienz zu steigern. Qlora [Det+23]
- Quantisierung - Mittels Quantisierung kann ein Modell auf eine geringere Bitbreite transformiert werden. Dies führt zu einer Effizienzsteigerung und erlaubt es darüber hinaus ein Modell auf Hardware Plattformen mit geringerer Bitbreite zu nutzen. Der damit einhergehende Genauigkeitsverlust bei den Werten der einzelnen Parametern kann zu schlechteren Inferenzergebnissen führen. [Fra+23]
- Activation aware Weight Quantization (AWQ) [Lin+24]

1.3 Einleitung - Auswahl des Modells

Als geeignetes LLM wird das Modell LLama 2 ausgewählt, dass am im Juli 2024 von der Firma Meta AI veröffentlicht wurde [Tou+23]. Das Modell wurde unter der Open-Source-Lizenz GPL3 (Zitat) veröffentlicht und liegt in drei Varianten mit unterschiedlicher Parameteranzahl vor. Für diese Arbeit wird die kleinste der drei verfügbaren Varianten mit sieben Milliarden Parametern ausgewählt. Neben dieser existieren noch zwei weitere Varianten des Modells, eine mit dreizehn und eine mit siebzig Milliarden Parametern. Die Auswahl des kleinsten Modells erfolgt vor allem unter dem Gesichtspunkt der notwendigen Ressourcen und des damit einhergehenden zeitlichen Aufwands, für die verschiedenen zu untersuchenden Verfahren. Bei größeren Modellen ist davon auszugehen, dass die Verfahren zur Effizienz Optimierung deutlich mehr Zeit benötigen. Es wird erwogen, die an dem kleineren Modell erprobten vielversprechendsten Strategien in einem zweiten Schritt, auch auf die größeren Varianten des LLama 2 Modells zu übertragen.

1.4 Ansatz für die Analyse

Die Analyse soll auf dem FH-Cluster der Fachhochschule Iserlohn für Machine Learning erfolgen. Erste Schritte zur Umsetzbarkeit wurden bereits unternommen und es konnten bis jetzt keine Probleme identifiziert werden. Folgende Schritte

- Auswahl bzw. Entwicklung eines geeigneten Testszenarios
- Einzelanalyse der Effizienz Verfahren ggf. mit unterschiedlicher Parametrisierung
- Kombination der Technologien zur weiteren Effizienzsteigerung
- Vergleich der Ergebnisse von Einzel und Kombinations-Analyse
- Anwendung auf die größeren Varianten des LLama Modells

1.5 Auswertung

In der Auswertung sollen die Ergebnisse der einzelnen Technologien und der Kombination von Verfahren gegen das Ursprungsmodell verglichen werden. Insbesondere sollen auch die sinnvollen Grenzen der Effizienzsteigerung aufgezeigt und der Erwartungshorizont (ggf. anhand von Beispielen) skizziert werden. Da das AI Umfeld einer schnellen Wandlung unterliegt, soll am Ende

der Arbeit noch ein Status zu ggf. neuen oder verbesserten Technologien zur Effizienzsteigerung erfasst werden. Ebenfalls soll eine Einschätzung/Abgrenzung im Bezug zu ähnlichen Arbeiten in diesem Umfeld, abhängig von den Ergebnissen weiterer Recherche, vorgenommen werden. Abschießend erfolgt ein Zusammenfassung mit dem Ausblick auf den weiteren Verlauf in diesem Forschungsfeld.

2 Ergebnisse der Experimente

2.1 Optimierungstechnologien für LLMs

In dem folgenden Kapitel werden die Ergebnisse zu den verschiedenen Verfahren der Modell Optimierung vorgestellt. Eine Übersicht bietet die folgende Tabelle.

Verfahren	Llama 3 8B Instruct	Mistral 7B Instruct v0.2	Llama 2 7B chat
Low rank adaption (Lora)	X	X	X
Quantized low rank adaption (Qlora)	X	X	X
Activation-aware Weight Quantization (AWQ)	X	X	X
Depth pruning (Layer Extraction)	X	X	X
Knowledge distillation	X	X	X
Kombinierte Verfahren			
Pruning + Lora	X	X	X
Pruning + Lora + AWQ	X	X	X

Tabelle 1: Übersicht über die verwendeten Verfahren und Modelle

Für eine Verringerung des Speicherbedarfs eignen sich bevorzugt die Verfahren Quantization und Pruning, während Lora und Knowledge Distillation für das erneute Tuning solcher Modelle eingesetzt werden.

Quantization

Benchmark Ergebnisse der AWQ 4bit Quantization im Vergleich zum jeweiligen Referenz Modell.

Benchmarks:	winogrande	truthfulqa_mc2	hellaswag	arc_challenge	Durchschnitt
Mistral 7 Referenz	0,7395	0,6682	0,8365	0,5606	0,7012
Standard Fehler	*0,0123*	*0,0152*	*0,0037*	*0,0145*	
Mistral 7 AWQ	0,7403	0,6752	0,8310	0,5700	0,7041
Standard Fehler	*0,0123*	*0,0151*	*0,0037*	*0,0145*	
Llama 3 Referenz	0,7206	0,5165	0,7582	0,5674	0,6407
Standard Fehler	*0,0126*	*0,0152*	*0,0043*	*0,0145*	
Llama 3 AWQ	0,7356	0,5099	0,7532	0,5572	0,6390
Standard Fehler	*0,0124*	*0,0152*	*0,0043*	*0,0145*	
Llama 2 Referenz	0,6646	0,4532	0,7547	0,4420	0,5786
Standard Fehler	*0,0133*	*0,0156*	*0,0043*	*0,0145*	
Llama 2 AWQ	0,6456	0,4509	0,7481	0,4462	0,5727
	0,0134	*0,0156*	*0,0043*	*0,0145*	

Tabelle 2: Die Tabelle zeigt die quantisierten Modellen zusammen mit den korrespondierenden Referenzen. Für die Quantisierung wurde das Verfahren AutoAWQ verwendet.

Speicherbedarf der AWQ Modelle im Vergleich zur jeweiligen Referenz

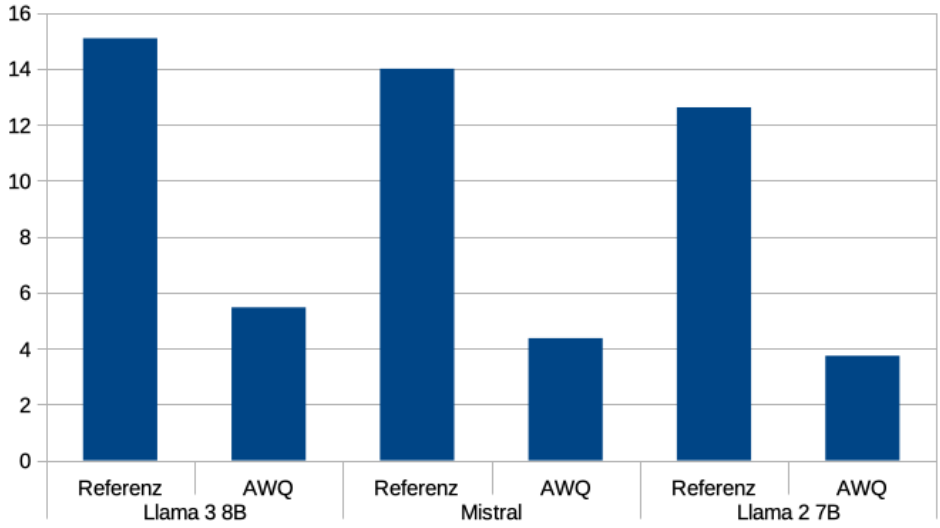

Abbildung 1: Auslastung des GPU Speichers der jeweiligen Modelle auf einer Nvidia A100 80GB GPU.

Pruning

Die Tabelle zeigt die Benchmark Ergebnisse für das Pruning der Referenz Modelle mittels PrunMe in jeweils zwei verschiedenen Ausprägungen

Benchmarks:	Layers Pruned	winogrande	truthfulqa_mc2	hellaswag	arc_challenge	Durchschnitt
Llama 3 8B	**0**	0,7206	0,5165	0,7582	0,5674	0,6407
	Standard Fehler	*0,0126*	*0,0152*	*0,0043*	*0,0145*	
	8	0,6369	0,5211	0,5897	0,4283	0,5440
	Standard Fehler	*0,0135*	*0,0159*	*0,0049*	*0,0145*	
	12	0,5706	0,5201	0,4204	0,3208	0,4580
	Standard Fehler	*0,0139*	*0,0164*	*0,0049*	*0,0136*	
Mistral 7 B	**0**	0,7395	0,6682	0,8365	0,5606	0,7012
	Standard Fehler	*0,0123*	*0,0152*	*0,0037*	*0,0145*	
	8	0,6811	0,6313	0,6646	0,4044	0,5954
	Standard Fehler	*0,0131*	*0,0158*	*0,0047*	*0,0143*	
	12	0,5943	0,5543	0,3239	0,3677	0,4601
	Standard Fehler	*0,0138*	*0,0167*	*0,0047*	*0,0141*	

Tabelle 3: Die Tabelle zeigt Modelle bei denen eine unterschiedliche Anzahl von Schichten mittels Pruning entfernt wurde.

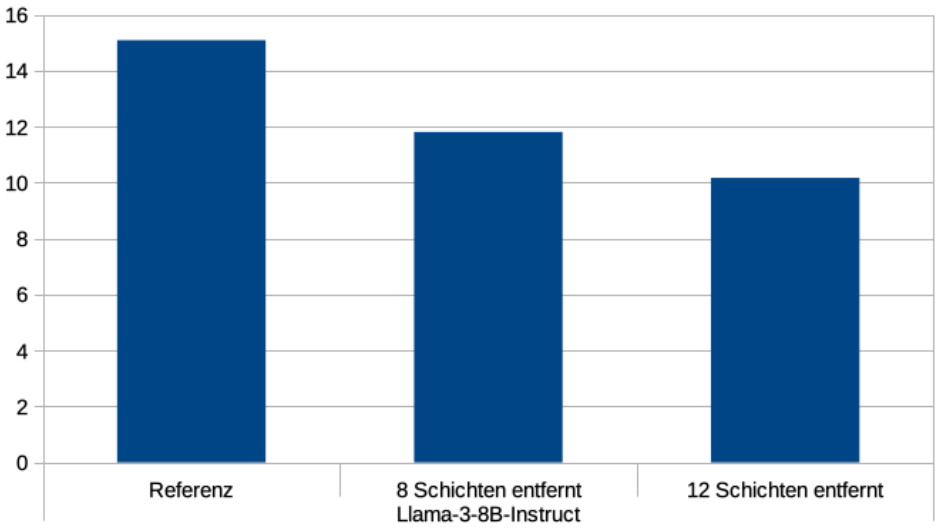

Abbildung 2: Auslastung des GPU Speichers der jeweiligen Modelle auf einer Nvidia A100 80GB GPU.

2.2 Kombination von Verfahren zur Kompression von LLM Modellen

Tuning eines geprunten Modells mittels Lora

Die Tabelle zeigt die Benchmark Ergebnisse für die Anwendung des Lora Verfahrens auf geprunte Modelle.

Modell	Verfahren	winogrande	truthfulqa_mc2	hellaswag	arc_challenge	Durchschnitt
Llama 3 8B Instruct	pruned	0,7111	0,5000	0,6255	0,4625	0,5748
		0,0127	*0,0157*	*0,0048*	*0,0146*	
	pruned + lora	0,7316	0,5189	0,7269	0,4974	0,6187
		0,0125	*0,0151*	*0,0044*	*0,0146*	
Mistral 7B Instruct v0.2	pruned	0,7072	0,6370	0,7298	0,4599	0,6335
		0,0128	*0,0158*	*0,0044*	*0,0146*	
	pruned + lora	0,7174	0,5774	0,7457	0,4855	0,6315
		0,0127	*0,0154*	*0,0043*	*0,0146*	
Llama 2 7B chat	pruned	0,6409	0,4955	0,6384	0,3968	0,5429
		0,0135	*0,0159*	*0,0048*	*0,0143*	
	pruned + lora	0,6732	0,4886	0,7082	0,4206	0,5727
		0,0132	*0,0154*	*0,0045*	*0,0144*	

Tabelle 4: Die Tabelle zeigt den Effekt der Anwendung des Lora Verfahrens auf die geprunten Modelle.

Tuning eines geprunten Modells mittels Knowledge Distillation

Einfluss von Knowledge Distillation auf die Ergebnisse.

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

Tabelle 5: Optimierungsversuch mit Knowledge Distillation auf Modelle die mit pruning verkleinert und mit lora wieder traniert wurden

Vergleichende Darstellung der erzeugten Varianten am Beispiel von Llama 3 8B Instruct

Die nachfolgende Tabelle zeigt die Benchmark Ergebnisse der zuvor beschrieben Llama 3 Modelle.

Benchmarks:	winogrande	truthfulqa_mc2	hellaswag	arc_challenge	Durchschnitt
Llama 3 8B instruct referenz	0,7210	0,5160	0,7580	0,5670	0,6400
Standard Fehler	*0,0130*	*0,0150*	*0,0040*	*0,0150*	
Llama 3 8B instruct pruned	0,7110	0,5000	0,6250	0,4620	0,5700
Standard Fehler	*0,0130*	*0,0160*	*0,0050*	*0,0150*	
Llama 3 8B instruct awq	0,7370	0,5099	0,7529	0,5580	0,6395
Standard Fehler	*0,0124*	*0,0152*	*0,0043*	*0,0145*	
Llama 3 8B instruct pruned + lora	0,7320	0,5190	0,7270	0,4970	0,6200
Standard Fehler	*0,0130*	*0,0150*	*0,0040*	*0,0150*	
Llama 3 8B instruct pruned + lora + awq	0,7230	0,5270	0,7320	0,4870	0,6200
Standard fehler	*0,0130*	*0,0150*	*0,0040*	*0,0150*	

Tabelle 6: Die Tabelle zeigt die Benchmark Ergebnisse der verschiedenen Llama 3 8B Instruct Modelle im Vergleich.

Perplexity Ergebnisse

Die nachfolgende Tabelle zeigt die Perplexity Ergebnisse für verschiedene vom Referenz Modell abgeleitet Varianten.

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
pruned	1,08	2,11	53,86
prune + lora	0,71	1,64	13,96
prune + lora + awq	0,70	1,62	13,24

Tabelle 7: Perplexity Messungen für die verschiedenen Varianten des LLama 3 8B Instruct Modells

Inference Ergebnisse

Die Tabelle zeigt die Inference Ergebnisse ermittelt auf einer Nvidia A100 80GB GPU (CUDA 11.7)

Modell	Referenz	awq*	pruned	lora	qlora	Pruned + lora	Pruned + lora +awq*
Llama-3-8B-Instruct	27,35	2,40	34,42	15,99	29,39	36,08	3,03
Mistral-7B-Instruct-v0.2	35,03	2,47	41,35	33,86	31,60	38,50	3,01
Llama-2-7b-chat	31.42	1.40	38.22	17.01	28.41	40.03	1.84

Tabelle 8: Inference Performance von nativen zu bearbeiteten Modellen. Die mit einem * gekennzeichneten Modell wurde in 4 Bit Genauigkeit geladen, da die verwendete T4 GPU nicht genug Speicherkapazität für die native Größe besaß.

Die Tabelle zeigt die Inference Ergebnisse ermittelt auf einer Nvidia T4 15GB GPU (CUDA 12.x)

Modell Name	Verfahren	Token pro Sekunde
Llama 3 8B	Referenz	10,22
	prune + lora + awq	33,34
Mistral-7B-Instruct-v02	Referenz	11,10
	prune + lora + awq	38,99

Tabelle 9: Inference Performance von nativen zu AWQ quantisierten Modellen. Die Messungen wurden in einer Google Colab Umgebung mit einer Nvidia T4 15 GB durchgeführt. Hier konnten Cuda 12.x Treiber genutzt werden, die eine Verwendung des AutoAWQ python Moduls ermöglichten.

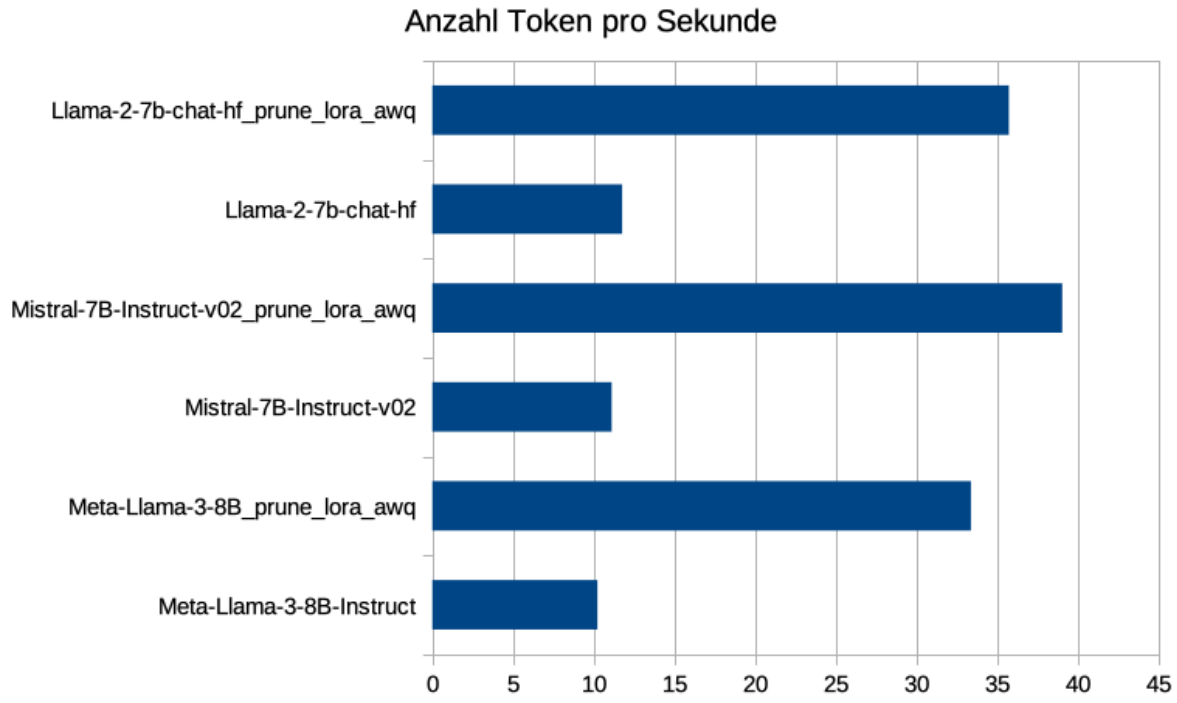

Abbildung 3: Token pro Sekunde ermittelt in einer Colab Umgebung mit CUDA 12.x und eine Nvidia T4 15GB GPU. Die Referenz Modelle wurden im 4bit mode geladen, da ansonsten der Speicher der GPU nicht ausgereicht hätte.

Systemresourcen

Speicherbedarf

Die Grafik zeigt die Speicherauslastung aller Varianten der drei Referenz Modelle

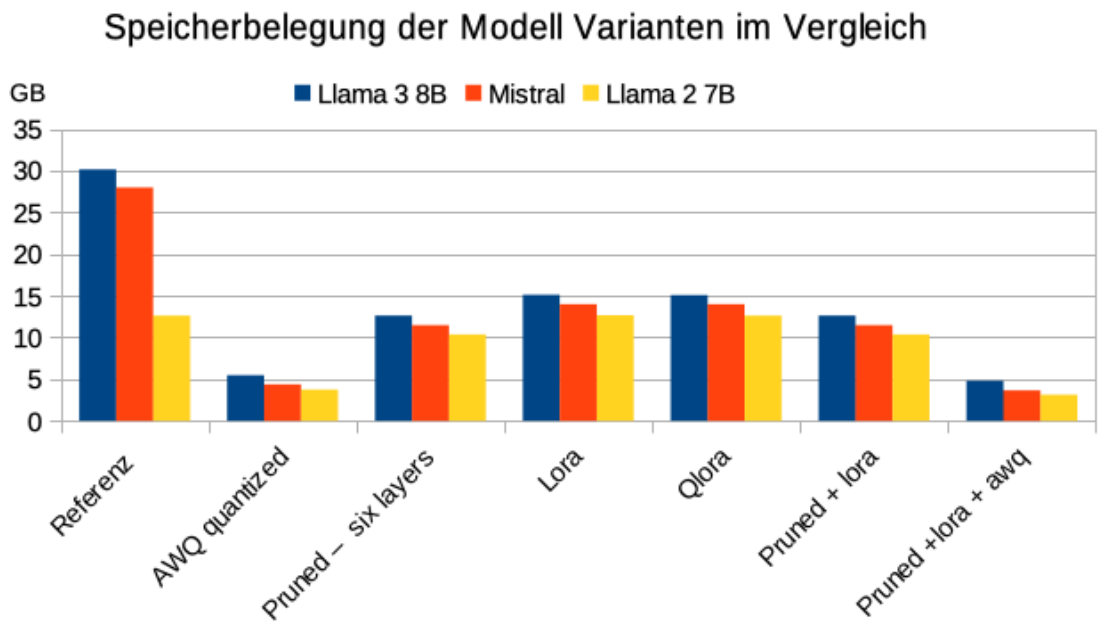

Abbildung 4: Speicherbedarf der Modelle auf einer Nvidia A100 80GB GPU. Aus dem jeweiligen Referenz Modell wurden alle weiteren Modelle abgeleitet. Die X-Achse zeigt die verwendeten Verfahren.

Literatur

[Det+23] Tim Dettmers u. a. *QLoRA: Efficient Finetuning of Quantized LLMs*. 2023. arXiv: 2305.14314 [cs.LG]. URL: https://arxiv.org/abs/2305.14314.

[FA23] Elias Frantar und Dan Alistarh. *SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot*. 2023. arXiv: 2301.00774 [cs.LG].

[Fra+23] Elias Frantar u. a. *GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers*. 2023. arXiv: 2210.17323 [cs.LG].

[Hu+21] Edward J. Hu u. a. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL].

[Kim+24] Bo-Kyeong Kim u. a. *Shortened LLaMA: Depth Pruning for Large Language Models with Comparison of Retraining Methods*. 2024. arXiv: 2402.02834 [cs.LG]. URL: https://arxiv.org/abs/2402.02834.

[Lin+24] Ji Lin u. a. *AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration*. 2024. arXiv: 2306.00978 [cs.CL]. URL: https://arxiv.org/abs/2306.00978.

[Sun+23] Mingjie Sun u. a. *A Simple and Effective Pruning Approach for Large Language Models*. 2023. arXiv: 2306.11695 [cs.CL].

[Tou+23] Hugo Touvron u. a. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288 [cs.CL].

[Xu+15] Yangyang Xu u. a. „Parallel matrix factorization for low-rank tensor completion“. In: *Inverse Problems Imaging* 9.2 (2015), S. 601–624. ISSN: 1930-8345. DOI: 10.3934/ipi.2015.9.601. URL: http://dx.doi.org/10.3934/ipi.2015.9.601.

[Xu+24] Xiaohan Xu u. a. *A Survey on Knowledge Distillation of Large Language Models*. 2024. arXiv: 2402.13116 [cs.CL].

[ZG17] Michael Zhu und Suyog Gupta. *To prune, or not to prune: exploring the efficacy of pruning for model compression*. 2017. arXiv: 1710.01878 [stat.ML].

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und dabei keine anderen als die angegebenen Hilfsmittel benutzt habe. Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Werken anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht. Die Arbeit wurde bisher weder gesamt noch in Teilen einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

1. August 2024

Dr. Thomas Schmitt