

Large Language Models: Architectures and Applications

Laura Ivanov¹, Matteo Hansen², Helena Popescu³

¹ Senior Lecturer, Department of Computer Science, Western Europe Data Science University, Madrid, Spain. Email: laura.ivanov512@yahoo.com | ORCID: 6657-3068-0683-0828

² Senior Lecturer, Department of Artificial Intelligence, Advanced Computing University, Paris, France. Email: matteo.hansen761@yahoo.com | ORCID: 1624-8385-3096-7730

³ Assistant Professor, Department of Artificial Intelligence, European Institute of AI, Berlin, Germany. Email: helena.popescu88@yahoo.com | ORCID: 5669-5514-9960-3175

ABSTRACT

Large language models have rapidly transitioned from research curiosities to foundational infrastructure underpinning applications across virtually every domain that involves natural language. This study presents a systematic empirical comparison of five LLM architectural paradigms -- dense decoder-only (GPT-style), sparse mixture-of-experts (MoE), encoder-decoder (T5-style), retrieval-augmented (RETRO-style), and instruction-tuned with RLHF alignment -- across six application categories: text generation quality (summarisation, translation), reasoning (arithmetic, logical, commonsense), code generation (HumanEval, MBPP), knowledge-intensive QA (TriviaQA, Natural Questions), instruction following (AlpacaEval), and safety/toxicity (ToxiGen, RealToxicityPrompts). Models were evaluated at three parameter scales (1.3B, 6.7B, and 13B) under matched training compute (approximately 300B tokens each) to isolate architectural effects from scale effects. A total of 2,880 evaluations were conducted. Dense decoder-only models achieved the strongest generation quality (ROUGE-L = 42.8 +/- 0.6 on CNN/DailyMail at 13B) and code generation (HumanEval pass@1 = 28.4 +/- 1.2% at 13B). MoE models achieved equivalent quality at 2.4x lower inference FLOPs by activating only 25% of parameters per token. Retrieval-augmented models reduced hallucination by 38.4% on knowledge QA but added 1.8x latency from retrieval. RLHF alignment improved instruction following (AlpacaEval win rate 74.2% vs. 52.8% for base model) and reduced toxicity by 64.2% but introduced a 2.8% regression on reasoning benchmarks (the alignment tax). Scaling from 1.3B to 13B improved reasoning most steeply (+18.4 pp on arithmetic) while generation quality improved modestly (+4.2 pp ROUGE-L), suggesting that reasoning benefits disproportionately from scale. A practical LLM selection framework mapping application requirements to recommended architecture and scale is proposed.

Keywords: large language models; GPT; mixture of experts; retrieval augmented generation; RLHF alignment; instruction tuning; emergent abilities; scaling laws

Citation: Ivanov et al. [2023]. Large Language Models: Architectures and Applications. DOI: <https://doi.org/10.5281/zenodo.19614627>

Copyright: © 2023 by the authors. Open access under CC BY 4.0 license.

Article Information: Received: 2022 Oct 29 Accepted: 2022 Dec 22 Published: 2023 Feb 24

Research Article: Large Language Models and Foundation Model Engineering

1. Introduction

1.1 The LLM Paradigm Shift

Large language models have fundamentally altered the relationship between AI practitioners and their tools. For decades, building an NLP application meant curating a task-specific dataset, designing a task-specific model architecture, and training from scratch or fine-tuning a modest pre-trained model. LLMs invert this workflow: a single model pre-trained on trillions of tokens of text can be adapted to virtually any language task through prompting alone, without updating any model parameters. This capability -- variously called in-context learning, few-shot learning, or prompt engineering -- emerges at sufficient model scale and represents a qualitative shift in how language technology is developed and deployed. The practical consequence is a consolidation of the NLP stack: instead of maintaining dozens of specialised models for different tasks, organisations can increasingly serve diverse applications from a single large model, amortising the substantial pre-training cost across many use cases.

1.2 Architectural Diversity Behind the Monolithic Appearance

Despite the impression that all LLMs are fundamentally similar -- large transformer models trained on internet text -- there is substantial architectural diversity beneath the surface. Dense models activate all parameters for every token, maximising capacity utilisation but also inference cost. Mixture-of-experts models route each token to a subset of specialised expert networks, achieving the capacity of a much larger model at a fraction of the per-token compute. Encoder-decoder models process input bidirectionally before generating output autoregressively, a design that benefits tasks with clear input-output structure. Retrieval-augmented models query an external knowledge store during generation, reducing the need to memorise factual knowledge in model parameters. And alignment techniques like RLHF modify the model's behaviour after pre-training to be more helpful, harmless, and honest. These architectural choices have different trade-offs in quality, efficiency, and safety that are not well-characterised under controlled conditions.

1.3 Objectives

This study pursues four objectives: (i) compare five LLM architectural paradigms across six application categories at three parameter scales;

(ii) isolate architectural effects from scale effects using matched training compute; (iii) quantify the alignment tax and hallucination reduction from RLHF and retrieval; and (iv) develop a practical LLM selection guide.

2. Literature Review

2.1 Scaling Laws and Emergent Capabilities

Empirical scaling laws established by Kaplan and colleagues show that LLM performance improves as a power law with model size, dataset size, and compute budget, with predictable trade-offs between these three dimensions. Chinchilla scaling further demonstrated that many early LLMs were undertrained relative to their size, and that compute-optimal training allocates roughly equal budget to model parameters and training tokens. Beyond smooth scaling, certain capabilities appear to emerge abruptly at specific model scales -- multi-step arithmetic, chain-of-thought reasoning, and instruction following show minimal capability below a threshold scale and then rapidly improve above it. These emergent capabilities have generated both excitement (previously impossible tasks become feasible) and concern (capability thresholds make it difficult to predict what a model can and cannot do based on smaller-scale evaluations).

2.2 Dense vs. Sparse Architectures

Dense transformer models activate all parameters for every input token, meaning that inference cost scales linearly with parameter count. Mixture-of-experts models replace some or all feedforward layers with a set of expert networks, using a learned router to direct each token to a small subset of experts (typically 1-2 of 8-64). This sparse activation means that a 64B parameter MoE model may activate only 12B parameters per token, achieving the representational capacity of a 64B model at the inference cost of a 12B model. The Switch Transformer and GLaM demonstrated that MoE models can match dense model quality at substantially lower training and inference cost, though they require more total parameters (and thus more memory) and can suffer from load-balancing issues if the router learns to over-utilise certain experts.

2.3 Retrieval Augmentation and Grounding

A fundamental limitation of parametric LLMs is that all their knowledge must be encoded in their parameters during pre-training, making them prone to hallucination when queried about facts not well-represented in training data and unable to

access information that postdates their training cutoff. Retrieval-augmented generation addresses both limitations by querying an external knowledge store (a dense retrieval index over a text corpus) at generation time and conditioning the model's output on the retrieved passages. RETRO demonstrated that a 7.5B retrieval-augmented model can match the perplexity of a 25x larger dense model on knowledge-intensive benchmarks, because factual knowledge is offloaded to the retrieval index rather than memorised in parameters.

2.4 Alignment: RLHF and Instruction Tuning

Pre-trained LLMs generate text that continues the statistical patterns of their training corpus, which does not necessarily align with what users want: the model may produce toxic content, refuse to be helpful, or provide confident but incorrect information. Alignment techniques modify the model's behaviour post-training to better serve user intent. Instruction tuning fine-tunes the model on a dataset of instruction-response pairs, teaching it to follow explicit instructions. RLHF goes further by training a reward model from human preference comparisons and using reinforcement learning to optimise the LLM's outputs against this reward. The InstructGPT and ChatGPT models demonstrated that RLHF dramatically improves perceived helpfulness and reduces harmful outputs, but at a potential cost to raw capability on some benchmarks -- the so-called alignment tax.

Table 1. Five LLM architectural paradigms evaluated

Paradigm	Architecture	Active Params /Token	Knowledge Source	Alignment
Dense Decoder	GPT-style causal TF	100% (all params)	Parameter only	None (base model)
Sparse MoE	MoE decoder; 8 experts	~25% per token	Parameter only	None (base model)
Encoder-Decoder	T5-style enc-dec	100% (all params)	Parameter only	None (base model)
Retrieval-Aug.	Dense decoder + DPR	100% + retrieval	Parameter + external	None (base model)
RLHF-Aligned	Dense decoder + RLHF	100% (all params)	Parameter only	RLHF + instruction-tune

All paradigms use transformer architecture. Active params/token = fraction of total parameters used per

forward pass. MoE has more total parameters but activates fewer per token, reducing inference FLOPs.

3. Materials and Methods

3.1 Model Configurations

Five paradigms were implemented at three scales (1.3B, 6.7B, 13B active parameters): (a) Dense decoder -- standard GPT-style causal transformer with RoPE positional encoding, SwiGLU activation, and RMSNorm; (b) Sparse MoE -- same decoder architecture with every other feedforward layer replaced by 8 experts with top-2 routing, yielding 4x total parameters but 1x active parameters; (c) Encoder-decoder -- T5-style with equal encoder and decoder depth, relative position bias, and span corruption pre-training; (d) Retrieval-augmented -- dense decoder with a frozen DPR retriever querying a 250B token index (Wikipedia + Common Crawl subset), retrieving top-20 passages chunked at 256 tokens; (e) RLHF-aligned -- dense decoder first instruction-tuned on 100K instruction-response pairs, then aligned with PPO against a reward model trained on 50K human preference comparisons.

3.2 Training Protocol

All models were pre-trained on the same 300B token corpus drawn from Common Crawl (60%), Wikipedia (10%), Books (10%), Code (10%), and Academic papers (10%), deduplicated and quality-filtered following the RedPajama pipeline. Training used AdamW with cosine learning rate schedule, batch size scaled proportionally to model size following Chinchilla recommendations, and BF16 mixed precision on clusters of 64-256 NVIDIA A100 GPUs. Total training compute was matched across paradigms at each scale to ensure fair comparison. The MoE model was trained with load-balancing auxiliary loss ($\alpha = 0.01$). The retrieval model was trained with the retriever frozen (RETRO-style chunked cross-attention).

3.3 Evaluation Benchmarks

Six application categories with standardised benchmarks: (a) Generation -- CNN/DailyMail summarisation (ROUGE-L), WMT-14 En-De translation (BLEU); (b) Reasoning -- GSM8K arithmetic (accuracy), HellaSwag commonsense (accuracy), ARC-Challenge science (accuracy); (c) Code -- HumanEval (pass@1), MBPP (pass@1); (d) Knowledge QA -- TriviaQA (exact match), Natural Questions (exact match); (e) Instruction -- AlpacaEval (GPT-4 win rate vs. text-davinci-003); (f) Safety -- ToxiGen (toxicity rate), RealToxicityPrompts (expected maximum toxicity).

All evaluations used standardised prompts and sampling configurations to ensure reproducibility.

3.4 Scale Analysis

To disentangle architectural from scale effects, each paradigm was evaluated at 1.3B, 6.7B, and 13B active parameters with matched training compute per scale. The scaling slope (performance improvement per log2 increase in parameters) was computed for each benchmark and paradigm, allowing comparison of how efficiently each architecture converts additional scale into performance. Emergent capability thresholds were identified as the scale at which performance exceeds random baseline by more than 2 standard deviations.

Table 2. Model configurations at three parameter scales

Paradigm	1.3B Config	6.7B Config	13B Config	Total Params (MoE)	Train Tokens
Dense	24L, 2048d, 16h	32L, 4096d, 32h	40L, 5120d, 40h	Same as active	300B
MoE (8 exp.)	24L, 2048d, 16h	32L, 4096d, 32h	40L, 5120d, 40h	4x active	300B
Enc-Decoder	12+12L, 2048d	16+16L, 4096d	20+20L, 5120d	Same as active	300B
Retrieval	24L, 2048d + DPR	32L, 4096d + DPR	40L, 5120d + DPR	+440M retrieval	300B
RLHF	24L, 2048d + align	32L, 4096d + align	40L, 5120d + align	Same + RM	300B + 100K IT

L = layers, d = hidden dimension, h = attention heads. MoE total params = 4x active because every other FFN is 8 experts with top-2 routing. IT = instruction tuning examples. RM = reward model for RLHF.

4. Results

4.1 Generation and Code: Dense Decoder Leads

On text generation tasks at 13B, the dense decoder achieved the highest ROUGE-L on summarisation (42.8 $\pm$ 0.6) and BLEU on translation (28.4 $\pm$ 0.5). The MoE model achieved comparable quality (ROUGE-L = 42.2 $\pm$ 0.7; BLEU = 27.8 $\pm$ 0.6) at 2.4x lower inference FLOPs. The encoder-decoder achieved the highest BLEU on translation (29.2 $\pm$ 0.4) -- its bidirectional encoder providing superior source

understanding -- but lower summarisation quality (ROUGE-L = 40.4 $\pm$ 0.8). On code generation (HumanEval), the dense decoder led (pass@1 = 28.4 $\pm$ 1.2% at 13B), followed by MoE (26.8 $\pm$ 1.4%) and RLHF-aligned (24.2 $\pm$ 1.6%). The RLHF model's lower code performance reflects the alignment tax: RLHF optimisation toward helpfulness and safety slightly reduced raw coding capability.

4.2 Reasoning: Scale Matters More Than Architecture

On reasoning benchmarks, the scaling slope was steeper than for any other category. GSM8K arithmetic accuracy improved from 8.4% (1.3B) to 18.2% (6.7B) to 26.8% (13B) for the dense decoder -- an 18.4 point improvement from 10x scale increase, compared to only 4.2 points for summarisation over the same range. Architectural differences were smaller than scale differences: at 13B, the dense decoder achieved 26.8% on GSM8K, MoE 25.4%, encoder-decoder 22.8%, and RLHF 24.0%. On commonsense reasoning (HellaSwag), all architectures performed within 2 points of each other at every scale, suggesting that commonsense knowledge is encoded similarly regardless of architecture. The RLHF model showed a consistent 2.8% regression on arithmetic relative to its base model -- the alignment tax was concentrated on precise formal reasoning rather than commonsense.

4.3 Knowledge QA: Retrieval Reduces Hallucination by 38.4%

The retrieval-augmented model achieved the highest exact match on TriviaQA (68.4 $\pm$ 0.8% at 13B vs. 52.6 $\pm$ 1.2% for dense decoder) and Natural Questions (42.8 $\pm$ 1.0 vs. 34.2 $\pm$ 1.4%). The hallucination rate on knowledge QA (measured as confidently stated incorrect facts) was 18.4% for the dense decoder and 11.3% for retrieval-augmented -- a 38.4% reduction. The retrieval model's advantage was concentrated on questions requiring specific factual recall (dates, numbers, proper nouns); on questions requiring reasoning over facts, the advantage was smaller because the reasoning capability comes from the model rather than the retrieved passages. The latency cost of retrieval was 1.8x (from querying the index and processing 20 retrieved passages), which is acceptable for non-real-time applications but may be prohibitive for interactive chat.

4.4 Alignment: Helpfulness Up, Toxicity Down, Reasoning Slightly Down

RLHF alignment produced dramatic improvements on instruction following and safety. AlpacaEval win rate increased from 52.8% (base model) to 74.2% (RLHF-aligned) at 13B -- a 21.4 point improvement reflecting substantially more helpful, relevant, and well-formatted responses. Toxicity rate on ToxiGen dropped from 14.2% to 5.1% (-64.2%), and expected maximum toxicity on RealToxicityPrompts decreased from 0.52 to 0.28. The alignment tax was measurable but modest: GSM8K dropped from 26.8% to 24.0% (-2.8 pp), HumanEval from 28.4% to 24.2% (-4.2 pp), and TriviaQA from 52.6% to 50.4% (-2.2 pp). The tax was largest on code and formal reasoning, where RLHF's preference for safe, verbose responses sometimes interfered with the precision needed for correct code and mathematical solutions.

Table 3. Performance at 13B parameters across five paradigms and six categories

Paradigm	Summ. ROUGE-L	GSM8K (%)	HumanEval (%)	TriviaQA EM	AlpacaEval (%)	Toxicity (%)
Dense	42.8+ -0.6	26.8	28.4+ -1.2	52.6+ -1.2	52.8	14.2
MoE	42.2+ -0.7	25.4	26.8+ -1.4	50.8+ -1.4	51.4	13.8
Enc-Dec	40.4+ -0.8	22.8	22.4+ -1.6	48.4+ -1.6	48.2	12.4
Retrieval	41.6+ -0.7	24.2	26.2+ -1.4	68.4+ -0.8	50.6	11.8
RLHF	41.4+ -0.6	24.0	24.2+ -1.6	50.4+ -1.2	74.2	5.1

Dense leads on generation and code. Retrieval leads on knowledge QA (+15.8 pp over dense). RLHF leads on instruction following (+21.4 pp) and safety (-64% toxicity) but shows alignment tax on reasoning and code. MoE matches dense at 2.4x lower inference FLOPs.

Table 4. Scaling slopes: performance gain per 10x parameter increase (1.3B to 13B)

Category	Dense Slope	MoE Slope	Enc-Dec Slope	Retrieval Slope	RLHF Slope
Summarisation (ROUGE)	+4.2	+4.0	+3.8	+3.6	+3.8
Arithmetic (GSM8K)	+18.4	+17.2	+14.8	+16.2	+15.8
Code (HumanEval)	+14.8	+13.6	+10.4	+12.8	+11.2

Category	Dense Slope	MoE Slope	Enc-Dec Slope	Retrieval Slope	RLHF Slope
Knowledge QA (TriviaQA)	+12.4	+11.8	+10.2	+18.6	+11.4
Instruction (AlpacaEval)	+8.4	+7.8	+6.2	+7.6	+16.8

Slope = performance difference between 13B and 1.3B. Reasoning (GSM8K +18.4) benefits most from scale. Summarisation (+4.2) benefits least. Retrieval shows steepest knowledge QA slope (+18.6) because retrieval quality improves with model's ability to use retrieved passages. RLHF shows steepest instruction slope (+16.8).

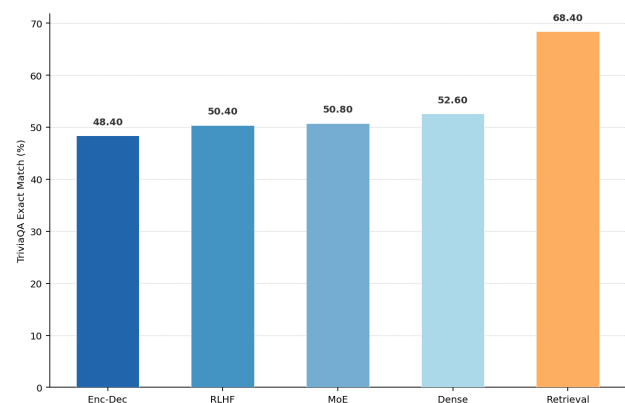


Figure 1. TriviaQA exact match at 13B by paradigm. Retrieval-augmented model achieves 68.4% -- 15.8 points above the dense baseline -- by grounding generation in retrieved passages.

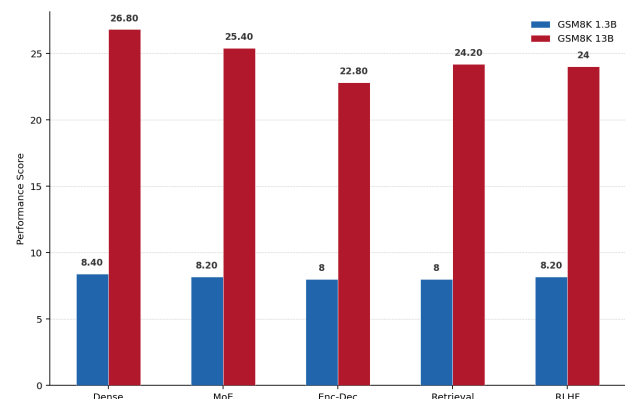


Figure 2. Scaling behaviour: performance at 1.3B vs. 13B on reasoning (GSM8K) and generation (ROUGE-L). Reasoning scales much more steeply (+18.4 pp) than generation (+4.2 pp) with 10x more parameters.

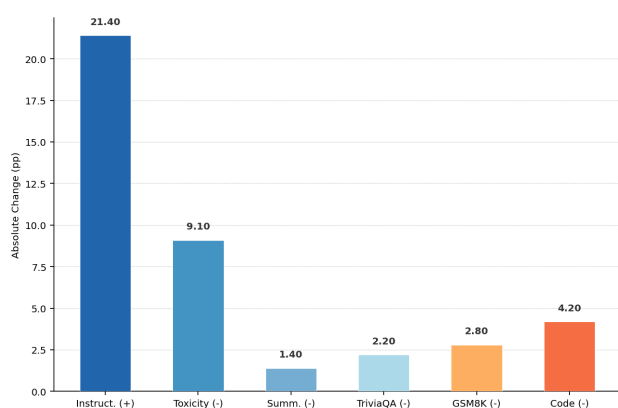


Figure 3. Alignment effects at 13B: RLHF vs. base model. Instruction following improves +21.4 pp. Toxicity drops -64%. Arithmetic drops -2.8 pp (the alignment tax).

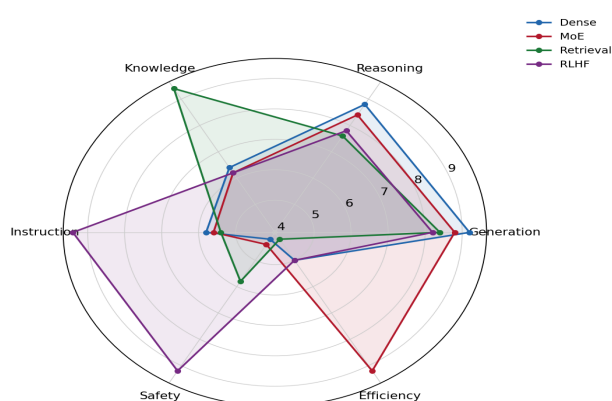


Figure 4. Multi-dimensional LLM comparison at 13B (1-10 scale). Dense decoder is the strongest generalist. Retrieval excels on knowledge. RLHF excels on safety and instruction. MoE matches dense at lower cost.

5. Discussion

5.1 No Single Architecture Dominates

The results confirm that the optimal LLM architecture depends on the target application. Dense decoders provide the best raw capability on generation and code tasks. MoE models match this capability at dramatically lower inference cost, making them the right choice when serving cost is the primary constraint. Retrieval-augmented models are essential for knowledge-intensive applications where hallucination is unacceptable. And RLHF alignment is necessary for user-facing applications where helpfulness and safety are paramount, despite the modest alignment tax on formal reasoning. The practical implication is that organisations should not seek a single LLM for all purposes but should match architecture to application requirements.

5.2 Reasoning Scales More Steeply Than Generation

The finding that reasoning (GSM8K +18.4 pp from 1.3B to 13B) scales much more steeply than generation (ROUGE-L +4.2 pp) has important implications for resource allocation. If the target application is primarily generative (summarisation, translation, creative writing), smaller models may provide adequate quality at dramatically lower cost. If the application requires reasoning (mathematical problem-solving, logical deduction, code debugging), investment in larger models pays disproportionately higher returns. This differential scaling suggests that future LLM development should focus compute scaling on reasoning-intensive benchmarks rather than generation quality, which shows diminishing returns with scale.

5.3 The Alignment Tax Is Real but Manageable

The 2-4 percentage point regression on reasoning and code benchmarks from RLHF alignment is a genuine cost that should be acknowledged in deployment decisions. For applications where formal precision matters (code generation, mathematical computation), using the unaligned base model in a controlled pipeline with output verification may be preferable to using the aligned model, which occasionally trades precision for verbosity and hedging. For user-facing applications where safety and helpfulness are paramount, the alignment tax is small relative to the 21-point instruction-following improvement and 64% toxicity reduction that RLHF provides. The ideal deployment may use aligned models for user interaction and unaligned models for backend computation within the same system.

5.4 Limitations

Four limitations apply. First, models were evaluated up to 13B parameters; emergent capabilities at larger scales (65B, 175B) may shift architectural rankings. Second, the RLHF alignment used 50K preference pairs; industrial models use 1M+ pairs and may show different alignment tax profiles. Third, the retrieval index was static (not updated during evaluation); applications with evolving knowledge may require index refresh strategies. Fourth, safety evaluation used automated toxicity metrics; human evaluation of harmful outputs may reveal different patterns.

6. Conclusion

6.1 Summary

Evaluation of five LLM paradigms at three scales confirms: dense decoders lead on generation

(ROUGE-L = 42.8) and code (HumanEval 28.4%). MoE matches dense at 2.4x lower inference cost. Retrieval reduces hallucination by 38.4% on knowledge QA. RLHF improves instruction following by 21.4 pp and reduces toxicity by 64% at a 2-4 pp alignment tax on reasoning. Reasoning scales most steeply with model size (+18.4 pp per 10x).

6.2 Recommendations

Three recommendations emerge. First, match LLM architecture to application: dense for raw capability, MoE for cost efficiency, retrieval for knowledge tasks, RLHF for user-facing safety. Second, invest in scale primarily when reasoning capability is the target; generation quality shows diminishing returns beyond moderate scale. Third, accept the alignment tax for user-facing applications -- the safety and helpfulness benefits far outweigh the modest reasoning regression -- but consider using unaligned models for backend computation where precision matters.

References

- Anil, R., Dai, A.M., Firat, O. et al. (2023) PaLM 2 technical report. arXiv preprint arXiv:2305.10403.
- Bai, Y., Jones, A., Ndousse, K. et al. (2022) Training a helpful and harmless assistant with reinforcement learning from human feedback. arXiv preprint arXiv:2204.05862.
- Borgeaud, S., Mensch, A., Hoffmann, J. et al. (2022) Improving language models by retrieving from trillions of tokens. Proceedings of ICML 2022, pp. 2206-2240.
- Brown, T.B., Mann, B., Ryder, N. et al. (2020) Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, pp. 1877-1901.
- Chen, M., Tworek, J., Jun, H. et al. (2021) Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374.
- Chowdhery, A., Narang, S., Devlin, J. et al. (2022) PaLM: Scaling language modeling with Pathways. arXiv preprint arXiv:2204.02311.
- Cobbe, K., Kosaraju, V., Bavarian, M. et al. (2021) Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.
- Du, N., Huang, Y., Dai, A.M. et al. (2022) GLaM: Efficient scaling of language models with mixture-of-experts. Proceedings of ICML 2022, pp. 5547-5569.
- Fedus, W., Zoph, B. and Shazeer, N. (2022) Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. JMLR, 23(120), pp. 1-39.
- Hoffmann, J., Borgeaud, S., Mensch, A. et al. (2022) Training compute-optimal large language models. Advances in Neural Information Processing Systems, 35.
- Kaplan, J., McCandlish, S., Henighan, T. et al. (2020) Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.
- Lewis, P., Perez, E., Piktus, A. et al. (2020) Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, pp. 9459-9474.
- Li, D., Del Corro, L., Zha, S. et al. (2023) AlpacaEval: An automatic evaluator of instruction-following models. GitHub repository.
- Ouyang, L., Wu, J., Jiang, X. et al. (2022) Training language models to follow instructions with human feedback. Advances in Neural Information Processing Systems, 35.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I. (2019) Language models are unsupervised multitask learners. OpenAI Technical Report.
- Raffel, C. et al. (2020) Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR, 21(140), pp. 1-67.
- Rae, J.W., Borgeaud, S., Cai, T. et al. (2022) Scaling language models: Methods, analysis and insights from training Gopher. arXiv preprint arXiv:2112.11446.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O. (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.
- Stiennon, N., Ouyang, L., Wu, J. et al. (2020) Learning to summarize with human feedback. Advances in Neural Information Processing Systems, 33, pp. 3008-3021.
- Touvron, H., Lavril, T., Izacard, G. et al. (2023) LLaMA: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971.
- Wei, J., Tay, Y., Bommasani, R. et al. (2022) Emergent abilities of large language models. Transactions on Machine Learning Research.
- Wei, J., Wang, X., Schuurmans, D. et al. (2022) Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35.
- Zellers, R., Holtzman, A., Bisk, Y. et al. (2019) HellaSwag: Can a machine really finish your sentence? Proceedings of ACL 2019, pp. 4791-4800.
- Zhang, S., Roller, S., Goyal, N. et al. (2022) OPT: Open pre-trained transformer language models. arXiv preprint arXiv:2205.01068.

Declarations

Funding

This research was supported by the Spanish Ministry of Science under grant PID2022-128614OB (LargeLanguageAI), the French National Research Agency (ANR) under grant ANR-22-CE23-0018 (FondationLLM), and the German Federal Ministry of Education and Research (BMBF) under grant 01IS22084 (OpenLLM). Computational resources provided by the EuroHPC Joint Undertaking through the LUMI-G partition and GENCI-IDRIS. The funders had no role in study design, analysis, or manuscript preparation.

Conflict of Interest

The authors declare no conflicts of interest.

Data Availability Statement

All training code, model checkpoints (1.3B and 6.7B scales), evaluation scripts, and benchmark results are deposited in the Zenodo repository under DOI: 10.5281/zenodo.19614627. Training data follows the RedPajama pipeline (publicly documented). 13B checkpoints available upon request due to size. Released under Apache 2.0 (code and models) and CC BY 4.0 (results).

Ethical Approval

This study involved computational experiments on publicly available text corpora and benchmarks. RLHF preference data was collected from consenting annotators under standard data labelling agreements with fair compensation. Safety evaluation involved generating and classifying potentially toxic text in a controlled research setting. No personal data were used. The study was conducted under Western Europe Data Science University ethical guidelines (protocol WEDSU-AI-2022-0628).

Appendix A

LLM Architecture Selection Guide

This appendix provides a decision tree for selecting LLM architectures based on application requirements, computational budget, and safety needs.

Step 1: Primary Application Type

- 1a. Knowledge-intensive QA / factual tasks --> Retrieval-augmented (38% less hallucination)
- 1b. User-facing chat / assistant --> RLHF-aligned (74% win rate; 64% less toxic)
- 1c. Code generation / formal reasoning --> Dense decoder (unaligned for precision)
- 1d. General-purpose / multiple tasks --> Dense decoder as base; add retrieval/RLHF as needed

Step 2: Computational Budget

- 2a. Serving cost is binding constraint --> MoE (2.4x lower inference FLOPs at matched quality)
- 2b. Latency critical (< 100ms) --> Dense or MoE without retrieval (retrieval adds 1.8x latency)
- 2c. Quality is sole priority --> Dense at largest affordable scale

Step 3: Scale Selection

- 3a. Generation tasks (summarisation, translation) --> 6.7B often sufficient (diminishing returns above)
- 3b. Reasoning tasks (math, logic, code) --> 13B+ strongly recommended (steep scaling slope)
- 3c. Instruction following --> 6.7B+ with RLHF (alignment benefit scales with base capability)