Large Model Architectures and Inference Optimization: Designs, Normalization, Activations, and Attention Efficiency

October 25, 2025

1 Mainstream Architecture Comparison (GPT, LLaMA, Qwen, Mistral, Mixtral)

1.1 GPT Family: Decoder-Only Baseline

The GPT series popularized the decoder-only transformer as the default large language model (LLM) blueprint:

- **Scaling and capacity:** GPT-3 (175B) uses 96 layers with 12,288 hidden dimensions and dense attention; GPT-4 introduces larger parameter counts, mixture-of-experts components, and multimodal extensions.
- **Data mixture:** A diverse blend of web text, code, dialog, and curated knowledge sources provides robustness across domains.
- **Alignment tooling:** RLHF, preference optimization, and tool-calling interfaces sit atop the base model, emphasizing safety and controllability.

GPT remains the reference point for evaluating architectural innovations in openness, context length, and alignment.

1.2 LLaMA Lineage: Efficiency and Long Context

Meta's LLaMA, LLaMA 2, and LLaMA 3 line focus on efficiency without sacrificing quality:

- **RMSNorm and SwiGLU:** Switching from LayerNorm+GELU improves numerical stability and convergence speed.
- **Grouped-Query Attention (GQA):** Sharing key/value heads across groups reduces KV cache size and boosts inference throughput.
- **Open ecosystem:** Clean training data, permissive licensing, and LoRA-friendly checkpoints fuel rapid community adoption.

LLaMA's architectural choices inspired many derivative open models tailored to specific domains.

1.3 Qwen Series: Multilingual and Multimodal

Alibaba's Qwen family targets multilingual coverage, tool integration, and multimodality:

- **Hybrid vocabulary:** A 1512M-token vocabulary covering Latin, CJK, code tokens, and structured tool signatures ensures versatility.
- **Extended positional schemes:** RoPE extrapolation with dynamic scaling supports 32K+ token contexts.

- **Vision-Language support:** Qwen-VL pairs a vision encoder with the text decoder through gated fusion layers for document and chart understanding.

Qwen provides instruction- and tool-tuned variants for customer service, search, and code-generation scenarios.

1.4 Mistral and Mixtral: Compact Excellence and Sparse Experts

Mistral AI emphasizes high-quality performance within tight compute budgets:

- **Sliding window attention:** Mistral 7B mixes local and global attention patterns to control quadratic cost without losing context comprehension.
- **Efficient multi-query attention:** Combining multi-query and multi-head attention dramatically shrinks KV state while preserving quality.
- **Mixtral MoE:** Mixtral-8x7B activates the top-2 of 8 experts per token, delivering 45B total parameters with 12B active, achieving superior throughput and accuracy trade-offs.

These models are popular for edge inference, quantization, and custom fine-tuning workloads.

2 Normalization Schemes (LayerNorm, RMSNorm)

2.1 LayerNorm: Classical Choice

Layer normalization standardizes each token representation:

$$\mathrm{LayerNorm}(h) = \frac{h-\mu}{\sqrt{\sigma^2+\epsilon}} \odot \gamma + \beta, \tag{1}$$

with learnable scale γ and bias β. Advantages include stable gradients and compatibility with residual connections. Drawbacks:

- Additional mean subtraction and square-root operations.
- Sensitivity to mixed-precision under large-scale training.
- Extra parameters and runtime overhead at massive scale.

2.2 RMSNorm: Simplified Stability

Root mean square normalization discards the mean term:

$$\mathrm{RMSNorm}(h) = \frac{h}{\mathrm{rms}(h)} \odot \gamma, \qquad \mathrm{rms}(h) = \sqrt{\frac{1}{d}\sum_{i=1}^{d} h_i^2 + \epsilon}. \tag{2}$$

Key properties:

- **No bias term:** Fewer parameters and operations.
- **Numerical robustness:** Works well with FP16/BF16 training and very deep networks.
- **Pre-norm compatibility:** Applied before residual branches to promote gradient flow.

RMSNorm underpins LLaMA, Mistral, and other modern open-source models.

2.3 Alternative Normalizations

Other variants include DeepNorm (layer scaling to stabilize >1000 layers), ScaleNorm (fixed norm), and norm-free architectures that rely on carefully tuned residual scaling. Mixture-of-experts setups may add expert-specific normalization to balance outputs.

3 Activation Functions (GELU, SwiGLU)

3.1 GELU: Gaussian Error Linear Unit

The Gaussian Error Linear Unit is defined as:

$$\mathrm{GELU}(x) = x \cdot \Phi(x), \tag{3}$$

with a smooth approximation

$$\mathrm{GELU}(x) \approx 0.5x \left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right]\right). \tag{4}$$

GELU offers:

- Smoother gating than ReLU, improving convergence on NLP tasks.
- Compatibility with transformer residual structures.
- Reasonable compute cost with high-quality approximations.

It remains the default activation for many encoder and decoder models.

3.2 SwiGLU: Gated Enhancements

SwiGLU extends gated linear units with Swish activation:

$$\mathrm{SwiGLU}(x) = \mathrm{Swish}(xW_1) \odot (xW_2), \qquad \mathrm{Swish}(z) = z \cdot \sigma(z). \tag{5}$$

Advantages:

- **Dynamic gating:** Learns multiplicative interactions to better allocate capacity.
- **Empirical gains:** Lowers perplexity in PaLM, LLaMA, and other large models.
- **Parameter-efficient scaling:** Often paired with wider hidden dimensions to maintain expressive power.

Costs include extra matrix multiplications and memory traffic, motivating fused implementations.

3.3 Activation Selection

Choosing between activations involves trade-offs:

- **Performance vs. efficiency:** SwiGLU is generally superior but heavier; GELU remains attractive for latency-sensitive deployments.
- **Precision considerations:** Saturating activations may require careful loss-scaling in FP16/BF16 settings.
- **Task alignment:** Generation-focused models benefit from SwiGLU's expressiveness, while understanding models may prioritize efficiency.

Benchmarking perplexity, throughput, and memory use guides final decisions.

4 FlashAttention and KV Cache Optimization

4.1 FlashAttention: IO-Aware Attention

FlashAttention reduces memory bandwidth bottlenecks while computing exact attention:

- **Tiling and recomputation:** Attention matrices are processed in tiles that fit on-chip SRAM, eliminating large intermediate tensor writes.
- **Fused kernels:** Softmax, scaling, and value weighting are fused into a single kernel for enhanced efficiency.
- **Numerical safety:** Online softmax with running maxima maintains stability despite tiling.

FlashAttention v2 generalizes the approach to multi-query, grouped attention, and non-square matrices, integrated via CUDA or Triton backends.

4.2 KV Cache Management and Compression

Autoregressive inference stores past key/value pairs to avoid recomputation:

- **Paged storage:** Partition KV buffers to reduce fragmentation and enable streaming between GPU and host memory.
- **GQA/MQA:** Grouped-query or multi-query attention share keys/values across heads, shrinking cache size by up to 8x.
- **Compression:** Quantize KV tensors to INT8/FP8 or apply low-rank projection to limit VRAM usage while preserving accuracy.

These optimizations deliver substantial latency and throughput gains for long-form generation.

4.3 Long-Context Inference and Retrieval Integration

Serving 100K+ contexts requires coordination across model, runtime, and data layers:

- **Positional extrapolation:** Techniques such as NTK scaling, XPos, and dynamic ALiBi maintain attention quality beyond training lengths.
- **Sliding windows and summarization:** Maintain a rolling window or compress stale history into summaries to bound cache growth.
- **Retrieval augmentation:** Retrieve relevant chunks from external stores to reduce dependence on massive KV states while retaining fidelity.

Operationally, batching, request scheduling, and multi-tenant resource isolation are critical to exploit these optimizations in production.

Further Reading

- Vaswani et al. “Attention Is All You Need.” NeurIPS, 2017.
- Touvron et al. “LLaMA: Open and Efficient Foundation Language Models.” arXiv, 2023.
- Jiang et al. “Mistral 7B.” arXiv, 2023.
- Dao et al. “FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness.” NeurIPS, 2022.
- Shazeer. “Fast Transformer Decoding: One Write-Head is All You Need.” arXiv, 2019.