LLMs are the culmination of decades of NLP, scaled with modern compute and data. Here's a practical path from core concepts to advanced techniques.

Foundations of language modeling

- Goal: Estimate the probability of a text sequence, often by predicting the next token.

$$ P(x) = \prod_{t=1}^{T} P(x_t \mid x_{<t}) $$

- Tokens: Subword units (e.g., BPE, WordPiece) balance vocabulary size and coverage.
- Embeddings: Dense vectors map tokens to continuous space for models to operate on.
- Decoding: Strategies like greedy, top-k, nucleus (top-p), and temperature control output diversity.

Neural networks for language

- From RNNs to Transformers: RNNs/LSTMs struggled with long contexts and parallelization; transformers solved both.
- Attention intuition: Learn "who to pay attention to" in the sequence for each token.
- Self-attention core:

$$ \text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V $$

- Multi-head attention: Multiple subspaces capture different relations (syntax, semantics, long-range dependencies).
- Positional encoding: Inject order via sinusoidal encodings or learned embeddings so attention knows token positions.

Transformer architecture and variants

- Base layers:
  - Self-attention: Context mixing per layer.
  - Feed-forward (MLP): Nonlinear transformation per token.
  - Normalization: LayerNorm for stable training.
  - Residuals: Skip connections to ease optimization.
- Decoder-only LLMs: Masked self-attention for causal generation (GPT-style).
- Encoder-decoder: Better for sequence-to-sequence tasks (translation, instruction following).
- Efficiency tweaks:
  - Sparse/linear attention: Reduce $\mathcal{O}(n^2)$ cost.
  - Rotary embeddings (RoPE): Improved positional handling.
  - Mixture-of-Experts (MoE): Conditional computation for scaling without linear cost.

Training objectives, scaling, and optimization

- Pretraining objective: Causal LM minimizes negative log-likelihood.

[\mathcal{L} = - \sum_{t=1}^{T} \log P_\theta(x_t \mid x_{<t})]

- Data: Diverse, deduplicated corpora; strong filters reduce contamination and toxicity.
- Scaling laws: Performance improves predictably with model size, data tokens, and compute; balance all three.
- Optimization:
  - Adam/AdamW: Adaptive optimizers with weight decay.
  - Learning rate schedules: Warmup + cosine decay are common.
  - Stability: Gradient clipping, careful init, mixed precision (FP16/BF16).
- Regularization: Dropout, weight decay, data augmentation (paraphrasing, masking).
- Curriculum & packing: Efficient batching and sequence packing improve throughput.

Alignment, fine-tuning, and control

- Supervised fine-tuning (SFT): Train on instruction–response pairs to shape behavior.
- RLHF:
  - Preference data: Human comparisons on outputs.
  - Reward model: Predicts which output is preferred.
  - Policy optimization: Optimize the LLM to maximize learned reward while preserving helpfulness/harmlessness.
- DPO/IPO: Direct preference optimization avoids online RL complexity.
- Parameter-efficient methods:
  - LoRA/Adapters: Inject low-rank updates; fine-tune cheaply.
  - Prefix/Prompt tuning: Learn soft prompts without touching core weights.
- Structured control: System prompts, tool use, function calling, persona conditioning.

Inference, prompting, and context management

- Prompting:
  - Zero/one-shot: Provide task/examples.
  - Chain-of-thought: Encourage step-by-step reasoning (use judiciously).
  - Self-consistency: Sample multiple rationales, pick consensus.
- Context windows: Larger windows enable longer documents but increase latency; use chunking and summaries.

- Decoding trade-offs:
    - Deterministic vs. diverse: Temperature and sampling tweak creativity vs. accuracy.
    - Logit biasing: Steer outputs away from unsafe or irrelevant tokens.
- Memory:
    - Short-term: In-prompt references.
    - Long-term: External stores plus retrieval for past interactions.

Retrieval-augmented generation and grounding

- Why RAG: Reduces hallucinations by injecting fresh, authoritative context.
- Pipeline:
    - Indexing: Chunk documents; create embeddings; store in a vector DB.
    - Retrieval: Hybrid search (BM25 + embeddings) improves relevance.
    - Augmentation: Build a compact context (map-reduce summaries, citations).
    - Generation: Constrain the LLM to answer from provided context.
- Enterprise fit: Policy Q&A, knowledge bases, semantic search, and auditability align with compliance needs.
- Evaluation: Measure grounded accuracy, citation coverage, context utilization, and latency.

Evaluation, robustness, and safety

- Quality metrics: Perplexity, exact match, F1, BLEU/ROUGE, faithfulness, factuality.
- Task-specific eval: MMLU, BigBench, domain tests; for enterprise, scenario-based user studies.
- Robustness: Adversarial prompts, jailbreak resistance, refusal calibration, red-teaming.
- Bias and fairness: Detect demographic skew; apply debiasing and policy filters.
- Privacy: PII detection, redaction, differential privacy in training if needed.

Efficiency, deployment, and operations

- Quantization: INT8/INT4 reduces memory and speeds up inference with minimal quality loss.
- Distillation: Train smaller student models from larger teachers for edge use.
- Serving:
    - Batching: Improves throughput at cost of latency.
    - Speculative decoding: Draft model accelerates generation.
    - Caching: KV-cache reuse for long prompts and streaming.
- Observability: Prompt/version tracking, drift detection, feedback loops.

- Guardrails: Content filters, policy checks, and citation requirements before final output.

Practical roadmap for you

- Core math & code:
    - Implement attention: Compute (Q,K,V), softmax, and multi-head aggregation.
    - Build a tiny transformer: Train on character-level data to see learning dynamics.
- Scale responsibly:
    - Tokenizer: Train BPE; experiment with vocab sizes.
    - Data pipeline: Deduplication, shard handling, curriculum.
- Enterprise focus:
    - RAG stack: Hybrid retrieval, chunking strategy, metadata filtering, policy grounding.
    - Evaluation: Define acceptance criteria (accuracy, latency, citation fidelity).
- Fine-tune: Try LoRA on domain documents; measure improvements in grounded Q&A.
- Ops: Add audit trails and safety checks aligned to compliance.