■■■ FIELD GUIDE // PUBLIC EDITION

AI Architecture, LLMs, Agents, Local AI, and Frontier Terms

Public Glossary and Quick Explainer

BlackHack Society . Home for Ethical Hackers

$ cat glossary.md

// public edition . for practitioners and the hacker-curious . version 2026.09

Eleven sections on AI and ML foundations through frontier governance, adversarial evaluation, and interpretability -- written for anyone learning how modern AI systems actually work.

Includes 23 charts, graphs, and workflow diagrams in the Visual Addendum, plus a deduplicated source index linking every claim to a primary reference.

-> Keep going -- see the next page for the BlackHack Academy Security+ track.

(c) 2026 BlackHack Society -- Home for Ethical Hackers. blackhacksociety.com . blackhackacademy.com . getblackhack.com

From the BHS AI Architecture Glossary, by BlackHack Society. Share it unaltered and with credit. No resale or derivative works without written permission.

■■■ CONTINUE THE JOURNEY

Keep going with BlackHack Society.

Start with our Security+ training track. Scan or tap to get started.

// CONTINUE THE JOURNEY

Security+ training track

Taught at BlackHack Academy through the Purple Team Mentality, in cohort-based training. Use the code below.

BLACKHACKSTART

20% off any BlackHack content. One use. No expiration.

SCAN OR TAP
blackhacksociety.com

SOCIETY
blackhacksociety.com
Community, events & speaking

ACADEMY
blackhackacademy.com
Purple Team Mentality, cohort training

STORE + EVENTS
getblackhack.com
Store and event resources

CompTIA and Security+ are trademarks of CompTIA, Inc., named here only to identify the exam. Nothing in this guide is legal, financial, or medical advice.

(c) 2026 BlackHack Society -- Home for Ethical Hackers. blackhacksociety.com . blackhackacademy.com . getblackhack.com
From the BHS AI Architecture Glossary, by BlackHack Society. Share it unaltered and with credit. No resale or derivative works without written permission.

▪▪▪ TABLE OF CONTENTS

What is inside

Numbered pointer to every section, appendix, and figure block. Page numbers are approximate — jump forward for the exact term list.

> Overview — why these terms matter operationally ··· p.3
> §1 Core AI and ML foundations ··· p.3
> §2 LLM basics ··· p.4
> §3 Transformer and model architecture ··· p.5
> §4 Local LLM and hardware ··· p.6
> §5 Retrieval, search, and knowledge grounding ··· p.8
> §6 Agent and tool-use ··· p.9
> §7 Protocols and ecosystem ··· p.10
> §8 Evaluation and benchmarks ··· p.10
> §9 Operational framing ··· p.11
> §10 Frontier, post-training, optimization, governance, bleeding-edge ··· p.12
> §11 Interpretability, oversight, and adversarial evaluation (new v2026.08) ··· p.16
> Field Notes — what matters operationally ··· p.19
> Source Index — every source cited in this glossary ··· p.20
> Join Us — Home for Ethical Hackers ··· p.22
> Visual Addendum — index (figures A01–A23) ··· p.23
> Original figures A01–A17 (from v2026.07) ··· p.24
> New figures A18–A23 (§11 explainers) ··· p.33
> Change Log ··· p.39

■■■ WHO WE ARE

BlackHack Society

Home for Ethical Hackers

BlackHack Society exists to raise the Next Generation of Ethical Hackers from Diverse Cultural Backgrounds. We operate a purple-team culture where offensive skill, defensive discipline, and civic responsibility train the same practitioner.

■■■ THE SIX PILLARS

> HONESTY > INTEGRITY

Say what is true. Flag what is not verified. Do the right work even when nobody is watching the terminal.

> AUTHENTICITY > HUMILITY

Show up as yourself. Culture is the moat. You are always one CVE away from being the target.

> ACCOUNTABILITY > GRATITUDE

Own the outcome, the incident, and the fix. Somebody opened the door for you. Hold it for the next hacker.

■■■ WHAT WE DO

SOCIETY — Civic and community face. Free clinics, workshops, and speaking. Trust is the product.

ACADEMY — Practitioner pipeline. Cohort-based training grounded in hacker culture and operated by hackers.

COMMUNITY — Aux Corps, Discord, events, and content. The place where hackers become each other's references.

■■■ WHY THIS MANUAL EXISTS

This is the shared vocabulary the BHS technical team uses when we plan curriculum, vet vendors, respond to AI-adjacent incidents, and ship agentic tooling. Every definition is one to three sentences. Every factual claim links to a primary or reference source you can defend in a review. Where a term is practitioner shorthand without a single canonical source, it is flagged in bold as **INFERENCE**.

■■■ OVERVIEW

Why these terms matter operationally

These terms show up in every AI conversation the team has now — hiring, procurement, compliance, incident response, curriculum, and product. Confusing a token window with a vector store, or an agent with a workflow, produces bad estimates, bad security postures, and bad bets. Use this document as the reference when you write specs, review vendor claims, or brief a partner.

■■■ SECTION 1

Core AI and ML foundations

The base vocabulary you need before any LLM or agent conversation. Get these wrong and the rest of the doc reads like noise.

Artificial intelligence (AI) — Technology that enables computers and machines to simulate human learning, comprehension, problem solving, decision-making, creativity, and autonomy. (IBM)

Machine learning (ML) — A subset of AI that creates models by training an algorithm to make predictions or decisions from data, enabling computers to learn and infer without being explicitly programmed for specific tasks. (IBM)

Deep learning — A subset of machine learning that uses multilayered ("deep") neural networks to more closely simulate complex human decision-making. (IBM)

Neural network — A machine learning model modeled loosely on the brain, consisting of interconnected layers of nodes (analogous to neurons) that work together to process and analyze complex data. (IBM)

Parameters — The internal numerical values a model learns during training; scale is often quoted in parameter counts (e.g., GPT-3's 175B parameters), and full fine-tuning retrains all of them. (LoRA (arXiv))

Weights — The learned parameters of a neural network; techniques like LoRA freeze pre-trained model weights and train small additional matrices instead of updating them all. (LoRA (arXiv))

Training data — The data an algorithm learns from; ML models learn from and make inferences based on data rather than explicit programming. (IBM)

Training — The process of creating a model by fitting an algorithm to data so it can make predictions or decisions; for LLMs this includes large-scale pre-training on general-domain data. (IBM), (LoRA (arXiv))

Inference — Running a trained model to produce outputs; for autoregressive Transformers, decoding K tokens takes K serial model runs, which makes inference slow. (Speculative decoding (arXiv))

Fine-tuning — A form of transfer learning that takes a pretrained model, adapts it to a target task/dataset, and adjusts it (often replacing or adding an output head) to particular tasks or domains. (Hugging Face)

Instruction tuning — Fine-tuning a model on demonstrations and rankings so it follows a user's intent; InstructGPT was fine-tuned with human feedback to better follow instructions, and a 1.3B InstructGPT was preferred over 175B GPT-3. (InstructGPT (arXiv))

RLHF (Reinforcement Learning from Human Feedback) — Aligning a model by first fitting a reward model to human preference labels, then fine-tuning the model with reinforcement learning to maximize that reward without drifting too far from the original. (DPO (arXiv)), (InstructGPT (arXiv))

LoRA (Low-Rank Adaptation) — A parameter-efficient fine-tuning method that freezes the pre-trained weights and injects trainable low-rank decomposition matrices into each Transformer layer, cutting trainable parameters (up to 10,000x for GPT-3) and GPU memory with no added inference latency. (LoRA (arXiv))

QLoRA — An efficient fine-tuning approach that back-propagates gradients through a frozen 4-bit quantized model into LoRA adapters, enabling fine-tuning of a 65B model on a single 48GB GPU while preserving 16-bit performance; it introduces the 4-bit NormalFloat (NF4) data type, double quantization, and paged optimizers. (QLoRA (arXiv))

■■■ SECTION 2

LLM basics

How a language model actually reads input, samples output, and where its behavior comes from at inference time.

Large language model (LLM) — A large pre-trained language model that stores factual knowledge in its parameters and achieves strong results across NLP tasks; making them bigger alone does not make them better at following user intent. (RAG (arXiv)), (InstructGPT (arXiv))

Token — A part of a sentence — usually a word, but also a subword or a punctuation symbol — used as the unit a model reads and generates. (Hugging Face)

Tokenization — The process of splitting text into tokens and mapping them to input IDs (numerical token indices) that the model consumes as input. (Hugging Face)

Context window — The maximum span of tokens a model can process at once; e.g., some OpenAI models have a ~1.05M-token context window with pricing that changes above 272K input tokens. (OpenAI pricing)

Prompt — The input text given to a model; function calling occurs when the model examines a prompt and decides it needs to call a tool to follow the instructions. (OpenAI docs)

System prompt — Instructions placed in the prompt to constrain model behavior; OWASP notes developers can build safeguards into system prompts, though such restrictions may be bypassed via prompt injection. (OWASP)

Zero-shot prompting — Prompting a model to do a task with no worked examples; chain-of-thought work contrasts this with providing few chain-of-thought demonstrations as exemplars. (Chain-of-Thought (arXiv))

Few-shot prompting — Providing a small number of demonstrations in the prompt; chain-of-thought prompting supplies "a few chain of thought demonstrations as exemplars in prompting." (Chain-of-Thought (arXiv))

Chain of thought (CoT) — Prompting a model to generate a series of intermediate reasoning steps, which significantly improves performance on complex arithmetic, commonsense, and symbolic reasoning tasks. (Chain-of-Thought (arXiv))

Hallucination — When a model produces untruthful or unsupported output; ReAct reduces "issues of hallucination and error propagation prevalent in chain-of-thought reasoning" by grounding actions in external sources. (ReAct (arXiv))

Sampling — A decoding strategy (multinomial sampling) that randomly selects the next token according to the probability distribution over the vocabulary, reducing repetition and producing more diverse output than greedy search. (Hugging Face)

Greedy decoding — The default decoding strategy that selects the single most likely token at each step; it works for short outputs but tends to repeat itself on longer sequences. (Hugging Face)

Beam search — A decoding strategy that keeps several candidate sequences (beams) at each step and finally selects the sequence with the highest overall probability, letting it "look ahead" beyond locally optimal tokens. (Hugging Face)

Temperature — A softmax scaling knob; lowering temperature makes the next-token distribution sharper (favoring high-probability words), and in the limit as temperature fi 0 it becomes equivalent to greedy decoding. (Hugging Face)

Top-p (nucleus) sampling — Sampling from the smallest set of words whose cumulative probability exceeds a threshold p, with the mass redistributed among them, so the candidate-set size adapts dynamically to the distribution. (Hugging Face)

Top-k — Sampling in which the K most likely next words are kept and the probability mass is redistributed among only those K words. (Hugging Face)

Logits — The raw, unnormalized scores a model outputs before softmax; inference servers apply "logits warpers" such as temperature scaling, top-p, top-k, and repetition penalty to them. (Hugging Face (TGI))

■■■ SECTION 3

Transformer and model architecture

The structural pieces that decide what a model can do, how fast it runs, and how it scales with data and compute.

Transformer — A network architecture based solely on attention mechanisms, dispensing with recurrence and convolutions, that is more parallelizable and trains faster than prior RNN/CNN encoder-decoder models. (Attention Is All You Need (arXiv))

Attention — A mechanism that connects elements of a sequence (originally the encoder and decoder) so the model can weight relationships between positions; it is the sole basis of the Transformer. (Attention Is All You Need (arXiv))

Self-attention — Attention within a single sequence, where each element of the input determines which other elements it should attend to. (Hugging Face)

Attention head — One of several parallel attention computations; multi-query attention shares keys and values "across all of the different attention 'heads.'" (Multi-Query Attention (arXiv))

Multi-head attention — Attention layers (as used in the Transformer) that run multiple attention heads in parallel to move information across and between sequences. (Multi-Query Attention (arXiv))

Multi-query attention (MQA) — A variant where the keys and values are shared across all attention heads, greatly reducing tensor size and the memory-bandwidth cost of incremental decoding with only minor quality loss. (Multi-Query Attention (arXiv))

Positional embeddings — Information added to Transformers to encode token order; position encoding enables dependency modeling between elements at different positions of the sequence. (RoFormer/RoPE (arXiv))

Rotary positional embeddings (RoPE) — A method that encodes absolute position with a rotation matrix while incorporating explicit relative-position dependency in self-attention, offering flexibility in sequence length and decaying inter-token dependency with distance. (RoFormer/RoPE (arXiv))

XPos / extended positional encoding — A length-extrapolatable position scheme (from the LEX Transformer) that uses a relative position embedding to maximize "attention resolution" plus blockwise causal attention at inference, so training on short texts generalizes to longer sequences. (Length-Extrapolatable Transformer (arXiv))

Encoder — The half of a sequence-transduction model that reads and represents the input; the original Transformer used an encoder-decoder configuration. (Attention Is All You Need (arXiv))

Decoder — The half that generates output tokens autoregressively; decoder/causal models generate one token at a time. (Hugging Face)

Encoder-decoder — An architecture pairing an encoder and a decoder (connected via attention in the best-performing pre-Transformer models), used for sequence-to-sequence tasks like translation. (Attention Is All You Need (arXiv))

LayerNorm — A normalization layer used in Transformers; the Switch Transformer showed large sparse models could be trained in lower-precision bfloat16 formats, relying on stable normalization. (Switch Transformer (arXiv)) (general normalization concept; see RMSNorm below for the LLM-common variant)

RMSNorm — Root Mean Square Layer Normalization, which regularizes the summed inputs to a neuron by their root mean square, giving re-scaling invariance while being computationally simpler and more efficient than LayerNorm (dropping LayerNorm's re-centering). (RMSNorm (arXiv))

Hidden dimension — The size of a model's internal representation (hidden state) that the model head projects onto a different dimension for a task. (Hugging Face)

Activation — The intermediate values flowing through network layers; tensor parallelism reduces "activation memory as the per-GPU tensor sizes shrink." (NVIDIA)

Expert routing — The routing algorithm in a Mixture-of-Experts model that selects which experts handle each input; the Switch Transformer "simplifies the MoE routing algorithm." (Switch Transformer (arXiv))

Mixture of Experts (MoE) — An architecture that selects different parameters (experts) for each input rather than reusing the same parameters, yielding a sparsely-activated model with very

many parameters but roughly constant compute per token. (Switch Transformer (arXiv))

Dense model — The conventional design where all parameters are used for every input, contrasted with sparsely-activated MoE models that use only selected parameters per example. (Switch Transformer (arXiv))

Open-weight model — A model whose trained weights are publicly released; the rise of open-access LLMs and fine-tuning methods like LoRA/PEFT (e.g., on Hugging Face) introduces new supply-chain risks. (OWASP)

Closed model — A proprietary model whose weights/training are not released; proprietary models may have unique training methods and source code considered sensitive, especially in closed or foundation models. (OWASP)

Model family — A set of related models sharing architecture/training but differing in scale (e.g., the QLoRA authors' Guanaco family, or LLaMA/T5 model types and scales). (QLoRA (arXiv))

Model size (7B, 14B, 70B, 1T) — The parameter count of a model, quoted in billions/trillions; the Switch Transformer pre-trained up to trillion-parameter models, and studies span 70M to 16B+ parameters. (Switch Transformer (arXiv)), (Chinchilla (arXiv))

Scaling laws — Empirical relationships between compute budget, model size, and training tokens that predict optimal training; the Chinchilla study trained 400+ models to derive them. (Chinchilla (arXiv))

Chinchilla-optimal / data-compute-parameter tradeoff — The finding that, for compute-optimal training, model size and training tokens should scale equally — for every doubling of model size, training tokens should also double; the resulting Chinchilla (70B, 4x more data) beat much larger models. (Chinchilla (arXiv))

▪▪▪ SECTION 4

Local LLM and hardware

What actually consumes memory and time on your own machines, and the levers that fit bigger models on smaller hardware.

GPU — A processor that accelerates ML by performing calculations in parallel; matrix-heavy operations see strong acceleration on GPUs. (NVIDIA)

CPU — The general-purpose processor; llama.cpp provides a plain C/C++ implementation with CPU support (AVX/AVX2/AVX512/AMX on x86) and CPU+GPU hybrid inference. (llama.cpp)

RAM — System memory used to hold model data and activations; half-precision (FP16) data reduces memory usage compared with FP32/FP64. (NVIDIA)

VRAM — GPU memory; LoRA reduced the GPU memory requirement ~3x versus full fine-tuning, and llama.cpp supports hybrid inference for models larger than total VRAM. (LoRA (arXiv)), (llama.cpp)

Bandwidth — The memory throughput between GPU memory tiers; FlashAttention is "IO-aware," minimizing reads/writes between GPU high-bandwidth memory (HBM) and on-chip SRAM. (FlashAttention (arXiv))

Quantization — Representing weights/activations in lower-precision integer formats (e.g., INT8, INT4) to speed up inference and reduce memory footprint and bandwidth. (NVIDIA)

FP32 — Single-precision (32-bit) floating point, a common float format in C-derived languages. (NVIDIA)

FP16 — Half-precision floating point, which reduces memory usage versus FP32/FP64 and makes data transfers faster, enabling larger networks. (NVIDIA)

BF16 — Brain floating point (bfloat16), a lower-precision format used, for example, by the Switch Transformer to train large sparse models for the first time in lower precision. (Switch Transformer (arXiv))

INT8 — 8-bit integer precision used for inference (e.g., CNNs), delivering roughly 3x+ speedups with reduced-precision compute. (NVIDIA)

INT4 — 4-bit integer precision (introduced with NVIDIA's Turing architecture) that adds further inference speedup and reduces memory footprint and bandwidth with minimal accuracy loss. (NVIDIA)

Q4 / Q5 / Q8 — llama.cpp integer quantization levels; it supports 1.5-, 2-, 3-, 4-, 5-, 6-, and 8-bit integer quantization for faster inference and reduced memory use. (llama.cpp)

GGUF — A binary file format optimized for quick loading/saving of models for inference (used with GGML/llama.cpp) that encodes both tensors and standardized metadata. (Hugging Face) llama.cpp — An open-source C/C++ LLM inference project whose main goal is LLM inference with minimal setup and state-of-the-art performance across a wide range of hardware, locally and in the cloud; it requires models in GGUF format. (llama.cpp)

Ollama — A lightweight, extensible framework for building and running language models on the local machine, with a REST API and support for importing GGUF models via a Modelfile. (Ollama)

LM Studio — A local application for running LLMs (Llama, DeepSeek, Qwen, Phi, etc.) on your own computer after downloading model weights, also exposing a REST API for local apps and scripts. (LM Studio)

Tokens per second (tok/s) — A throughput measure for generation; inference frameworks like TGI aim for increased total throughput via continuous batching. (Hugging Face (TGI))

Latency — The delay to produce output; SGLang is designed to deliver low-latency and high-throughput inference across single-GPU to distributed setups. (SGLang)

Throughput — The total volume served per unit time; PagedAttention/vLLM improves LLM throughput 2-4x at the same latency versus prior systems. (PagedAttention/vLLM (arXiv))

Batch size — The number of requests processed together; high-throughput serving requires batching many requests, but inefficient KV-cache memory management limits batch size. (PagedAttention/vLLM (arXiv))

Context length — The number of tokens processed per request; self-attention's time and memory complexity are quadratic in sequence length, making long contexts expensive. (FlashAttention (arXiv))

KV cache — The per-request key-value cache stored during generation; it is large and grows/shrinks dynamically, and mismanagement wastes memory through fragmentation and duplication. (PagedAttention/vLLM (arXiv))

Paged KV cache — KV-cache memory managed in pages (inspired by OS virtual memory/paging) so it can be allocated non-contiguously with near-zero waste. (PagedAttention/vLLM (arXiv))

PagedAttention — An attention algorithm inspired by classical virtual-memory paging that underpins vLLM, achieving near-zero KV-cache waste and flexible cache sharing within and across requests. (PagedAttention/vLLM (arXiv))

FlashAttention — An IO-aware exact attention algorithm that uses tiling to reduce memory reads/writes between GPU HBM and on-chip SRAM, requiring fewer HBM accesses than standard attention and enabling longer contexts. (FlashAttention (arXiv))

Sliding window attention — An attention pattern (used by Mistral 7B alongside grouped-query attention) that lets each token attend only to a fixed-size window of prior tokens, so the model can handle sequences of arbitrary length at reduced inference cost. (Mistral 7B (arXiv))

CPU offload — Splitting inference across CPU and GPU; llama.cpp supports "CPU+GPU hybrid inference to partially accelerate models larger than the total VRAM capacity." (llama.cpp)

Memory overhead — Extra memory consumed beyond weights; inefficient KV-cache handling wastes memory via fragmentation and redundant duplication. (PagedAttention/vLLM (arXiv))

VRAM formula for inference — Inference memory is dominated by model weights plus the KV cache, which grows with sequence length and batch size and can waste memory through fragmentation if unmanaged. (PagedAttention/vLLM (arXiv))

Training VRAM formula — Training memory is dominated by model state (parameters, gradients, optimizer states) plus activations; tensor parallelism reduces both model-state and activation memory by shrinking per-GPU tensor sizes. (NVIDIA)

Why training is harder than inference — Training must also store gradients and optimizer states and can hit memory spikes; QLoRA specifically adds paged optimizers to manage memory spikes and back-propagation through a 4-bit frozen model, and full fine-tuning of large models is often infeasible without such tricks. (QLoRA (arXiv))

Why context length increases memory demand — Self-attention's time and memory cost are quadratic in sequence length, and the KV cache stored per request grows with the number of tokens, so longer contexts consume more memory. (FlashAttention (arXiv)), (PagedAttention/vLLM (arXiv))

▪▪▪ SECTION 5

Retrieval, search, and knowledge grounding

How to give a model your data at query time so answers are grounded, not hallucinated.

Embedding — A vector representation of text that encodes meaning; an embedding model converts text chunks into vector embeddings for retrieval. (Anthropic)

Embedding model — A model that converts text into semantic vectors; embedding models excel at capturing semantic relationships but can miss exact matches. (Anthropic)

Vector — The numeric array (vector embedding) produced by an embedding model to encode a chunk's meaning for similarity search. (Anthropic)

Vector database — A store used at runtime to find the most relevant chunks based on semantic similarity to the query. (Anthropic)

Vector store — The index of chunk embeddings queried by semantic similarity; RAG's original non-parametric memory was a dense vector index of Wikipedia accessed by a neural retriever. (RAG (arXiv)), (Anthropic)

Semantic search — Retrieval that ranks chunks by semantic similarity to a query using embeddings, capturing meaning rather than exact words. (Anthropic)

Keyword search — Lexical matching to find exact word or phrase matches (e.g., via BM25), effective for unique identifiers or technical terms. (Anthropic)

BM25 — "Best Matching 25," a ranking function that uses lexical matching to find precise word/phrase matches, particularly effective for unique identifiers or technical terms. (Anthropic)

Hybrid retrieval — Combining BM25 (exact matches) and embeddings (semantic similarity) and fusing/deduplicating the results with rank-fusion, balancing precise term matching with broader semantic understanding. (Anthropic)

Chunking — Breaking a knowledge base into smaller text chunks (often no more than a few hundred tokens) for efficient retrieval. (Anthropic)

Reranking — A filtering step that reorders retrieved chunks so only the most relevant are passed to the model, improving responses and reducing cost and latency. (Anthropic)

Cross-encoder — A model that takes both texts simultaneously and outputs a single similarity score (0-1); it achieves higher accuracy than a bi-encoder but produces no reusable embeddings, so it is typically used to rerank a shortlist. (Sentence-Transformers)

Bi-encoder — A model that encodes each text independently into a sentence embedding, which can then be compared (e.g., by cosine similarity) and indexed for efficient retrieval. (Sentence-Transformers)

Metadata filtering — Restricting a vector search to records that match specified metadata conditions using operators like $eq, $gt, $in, $and, and $or, so retrieval can be scoped by tenant, source, date, or type. (Pinecone)

Retrieval-Augmented Generation (RAG) — Models that combine pre-trained parametric memory (the LLM) with non-parametric memory (a retrieved document index) for language generation, improving factuality and provenance on knowledge-intensive tasks. (RAG (arXiv))

Grounding — Providing external, retrieved context so outputs are supported by real sources; ReAct's actions interface with external knowledge bases/environments to gather information and reduce hallucination. (ReAct (arXiv))

Contextual embeddings — Anthropic's technique of prepending chunk-specific explanatory context to each chunk before embedding it, which reduced top-20 retrieval failures by 35%. (Anthropic)

▪▪▪ SECTION 6

Agent and tool-use

The vocabulary for systems where a model does not just answer, it takes actions and gets feedback.

Agent — A system where an LLM dynamically directs its own processes and tool usage, maintaining control over how it accomplishes tasks (as distinguished from fixed workflows); definitions

range from fully autonomous systems to prescriptive workflow-following implementations. (Anthropic)

Agent loop — The iterative cycle in which an agent acts and then gains "ground truth" from the environment at each step (tool results, code execution) to assess progress. (Anthropic)

Plan-act-reflect loop — The pattern of interleaving reasoning and acting: reasoning traces help induce, track, and update plans and handle exceptions, while actions gather information from the environment. (ReAct (arXiv))

ReAct pattern — Prompting an LLM to generate reasoning traces and task-specific actions in an interleaved manner, so reasoning guides planning and actions interface with external sources. (ReAct (arXiv))

Tool calling — A response in which the model, after examining a prompt, determines it must call one of the tools made available to it (also called function calling). (OpenAI docs)

Function calling — OpenAI's mechanism for models to interface with external systems and data; a function is a specific kind of tool defined by a JSON schema. (OpenAI docs)

Session — A bounded interaction context for pricing/serving; e.g., OpenAI applies long-context multipliers "for the full session" when input exceeds 272K tokens. (OpenAI pricing)

Runtime — The execution environment an agent interacts with; OSWorld provides a real computer environment supporting task setup, execution-based evaluation, and interactive learning. (OSWorld (arXiv))

Harness — The scaffolding/framework that runs a model against tasks; SWE-bench is an "evaluation framework" that tasks a model with editing a codebase to resolve real GitHub issues. (SWE-bench (arXiv))

Orchestrator — In the orchestrator-workers pattern, a central LLM that dynamically breaks down tasks, delegates them to worker LLMs, and synthesizes their results. (Anthropic)

Model router — **INFERENCE** — practitioner term; no single canonical source found. A routing layer in an application that picks which model (cheap vs. strong, fast vs. reasoning, local vs. cloud) should handle a given request; distinct from the intra-model expert routing used in Mixture-of-Experts. (Switch Transformer (arXiv))

Multi-agent system — Multiple AI agents working collectively to perform tasks on behalf of a user or system, where agents remain autonomous but cooperate and coordinate to achieve global goals. (IBM)

Swarm — A large multi-agent collective; multi-agent systems can encompass hundreds or thousands of coordinating agents to complete large-scale, complex tasks. (IBM)

Intra-domain coordination — Coordination among agents within a shared environment/domain, achieved directly or indirectly by altering the shared environment, with agents modeling each other's goals, memory, and plans. (IBM)

Cross-domain coordination — Coordinating agents across different platforms/providers; A2A gives a standardized method for managing agents across diverse platforms and cloud environments and combining agents from various providers. (Google Developers Blog)

Episodic memory — Memory that lets an agent recall specific past experiences (like individual events), useful for case-based reasoning. (IBM)

Semantic memory — Memory that stores structured, generalized factual knowledge — facts, definitions, and rules — that an agent retrieves for reasoning. (IBM)

Procedural memory — Memory that stores skills, rules, and learned behaviors so an agent can perform tasks automatically without explicit reasoning each time. (IBM)

Competence failure — **INFERENCE** — practitioner term; no canonical source located. A failure mode in which an agent produces a wrong answer or bad action not because of malice but because it lacks the skill, context, or grounding to succeed; contrasts with adversarial or Byzantine failures.

Soft Byzantine fault — **INFERENCE** — practitioner term; no canonical source located. A failure mode in a multi-agent system where a component doesn't crash but returns subtly wrong, inconsistent, or misleading outputs, echoing the classical Byzantine-fault problem in distributed systems.

Computer use — Giving a model screenshot capture plus mouse and keyboard control of a desktop environment for autonomous desktop interaction (e.g., Anthropic's computer use tool). (Anthropic)

OSWorld — A scalable, real computer environment and benchmark of 369 tasks across Ubuntu, Windows, and macOS for multimodal agents, with execution-based evaluation; humans solve 72%+ while the best model scored only 12.24% in the original study. (OSWorld (arXiv))

▪▪▪ SECTION 7

Protocols and ecosystem

The wires between apps, tools, and other agents. Standards here decide whether your stack is composable or locked in.

Model Context Protocol (MCP) — An open-source standard for connecting AI applications to external systems, providing tools and context to agents. (Model Context Protocol), (Google Developers Blog)

Agent-to-Agent (A2A) — Google's open protocol that lets AI agents communicate, securely exchange information, and coordinate actions across enterprise platforms; it complements MCP. (Google Developers Blog)

Agent Network Protocol (ANP) — An open-source communication protocol for intelligent agents that aims to be "the HTTP of the Agentic Web era," defining how agents connect into an open, secure, efficient collaboration network. (Agent Network Protocol)

Agent Control Protocol (ACP) — **INFERENCE** — no single canonical source located. A term used in the emerging agent-protocol landscape (alongside MCP, A2A, and ANP) for standardizing how a controller supervises, constrains, and revokes an agent's actions; verify against a specific implementation before adopting. (Agent Network Protocol)

API — A programmatic interface for models/services; OpenAI's function-calling lets models interface with external systems and access data outside their training via API calls, and providers expose REST APIs. (OpenAI docs), (Ollama)

API pricing per million tokens — Token-based billing where costs are quoted per 1M tokens, typically split into input, cached input, and output rates (e.g., gpt-5.4: $2.50 input / $0.25 cached input / $15.00 output per 1M tokens). (OpenAI pricing)

Prompt caching — Reusing cached prompt prefixes across requests to reduce processing time and cost for repetitive tasks; cached input tokens are billed at a lower rate (e.g., one-tenth of standard input for some OpenAI models). (Anthropic), (OpenAI pricing)

Anthropic-compatible endpoint — **INFERENCE** — practitioner term; no canonical source located. An HTTP endpoint that accepts the same request/response shape as Anthropic's Messages API so client code written for Claude can point at another provider (or a self-hosted server) with only a base-URL swap. (Anthropic)

▪▪▪ SECTION 8

Evaluation and benchmarks

How to tell whether a change actually made the system better. Also how to challenge vendor claims.

Eval — A test used to measure model capabilities; the field notes that "language models have outpaced our ability to evaluate them effectively." (SWE-bench (arXiv))

Benchmark — A standardized dataset/framework for comparing models, such as SWE-bench (2,294 real GitHub-issue problems) or MMLU. (SWE-bench (arXiv)), (MMLU (arXiv))

MMLU — "Measuring Massive Multitask Language Understanding," a test of multitask accuracy across 57 subjects (math, US history, computer science, law, and more) that probes world knowledge and problem solving. (MMLU (arXiv))

GPQA — A "Graduate-Level Google-Proof Q&A;" benchmark of 448 expert-written multiple-choice questions in biology, physics, and chemistry, designed to be extremely difficult even with web access. (GPQA (arXiv))

GPQA Diamond — A higher-quality, more challenging 198-question subset of GPQA where both expert annotators answered correctly but the majority of non-experts answered incorrectly. (Epoch AI)

SWE-Bench — An evaluation framework of 2,294 software-engineering problems drawn from real GitHub issues and pull requests across 12 Python repositories, tasking a model with editing a codebase

to resolve each issue. (SWE-bench (arXiv))

SWE-Bench Pro — A harder, contamination-resistant successor to SWE-Bench with 1,865 long-horizon, enterprise-grade software problems across 41 repositories (public, held-out, and commercial sets), often requiring multi-file patches that would take a professional engineer hours to days. (SWE-Bench Pro (arXiv))

Real-world workflow evaluation — Execution-based evaluation on realistic tasks; OSWorld derives each task from real-world computer use cases with custom execution-based evaluation scripts spanning multiple applications. (OSWorld (arXiv))

Reliability — For high-risk AI, the EU AI Act requires a "high level of robustness, cybersecurity and accuracy," and NIST's framework promotes trustworthiness in design and use. (EU AI Act), (NIST AI RMF)

Explainable AI (XAI) — Approaches that make model outputs and behavior understandable to humans; ReAct emphasizes "improved human interpretability and trustworthiness," and model documentation increases transparency into how AI works. (ReAct (arXiv)), (Model Cards (arXiv))

Bias — Systematic unfairness in outputs; model cards report benchmarked performance across cultural, demographic, and phenotypic groups to surface disparities, and vulnerable pre-trained models can carry hidden biases. (Model Cards (arXiv)), (OWASP)

Red teaming — Adversarial testing of AI systems; MITRE ATLAS is built on real-world attack observations and realistic demonstrations from AI red teams and security groups. (MITRE ATLAS)

Guardrails — Safety controls on inputs/outputs; developers can build safeguards into system prompts and input handling to mitigate prompt injection, though these can be bypassed. (OWASP)

Content filtering — Filtering harmful or sensitive outputs; adding restrictions in the system prompt about which data types the model may return can mitigate sensitive-information disclosure, but such restrictions may not be honored. (OWASP)

▪▪▪ SECTION 9

Operational framing

What each family of terms means for the six things a team has to defend: cost, security, performance, reliability, adoption, and local-vs-cloud tradeoffs.

Cost — LLM API cost is driven by token usage, billed per million tokens and split into input, cached-input, and output rates (e.g., gpt-5.4 lists $2.50 input, $0.25 cached input, and $15.00 output per 1M tokens), so long prompts and long outputs directly raise spend. (OpenAI pricing) Prompt caching reduces cost for repetitive prompts by reusing cached prefixes at a lower rate, and reranking cuts cost by passing fewer chunks to the model. (Anthropic), (Anthropic)

Security — AI-specific risks include prompt injection, where user inputs alter model behavior in unintended ways even if imperceptible to humans, and sensitive-information disclosure of PII or proprietary data through model output. (OWASP), (OWASP) MITRE ATLAS catalogs adversary tactics and techniques against AI-enabled systems based on real-world observations to help defenders. (MITRE ATLAS)

Performance — Serving performance is governed by throughput and latency; PagedAttention/vLLM improves throughput 2-4x at equal latency by eliminating KV-cache waste, and FlashAttention speeds attention by minimizing GPU memory I/O. (PagedAttention/vLLM (arXiv)), (FlashAttention (arXiv)) Continuous batching and tensor parallelism further raise total throughput and enable multi-GPU inference. (Hugging Face (TGI))

Reliability — Trustworthy AI requires managing risks throughout design, development, use, and evaluation, per the NIST AI Risk Management Framework, which is intended for voluntary use. (NIST AI RMF) For high-risk systems, the EU AI Act mandates risk assessment/mitigation, logging for traceability, human oversight, and a high level of robustness, cybersecurity, and accuracy. (EU AI Act)

Team adoption — Adoption is easier with standards and documentation: model cards give teams benchmarked performance and intended-use information to decide where a model fits, and protocols like MCP and A2A standardize how applications and agents connect to tools and to each other across platforms. (Model Cards (arXiv)), (Model Context Protocol), (Google Developers Blog)

Local vs cloud tradeoffs — Local runtimes such as Ollama, LM Studio, and llama.cpp let teams run models on their own machines with minimal setup, using quantization and CPU+GPU hybrid inference to fit models into limited VRAM. (Ollama), (LM Studio), (llama.cpp) Cloud APIs instead bill per token and can offer very large context windows, trading local control for managed scale and simpler ops. (OpenAI pricing)

▪▪▪ SECTION 10

Frontier, post-training, optimization, governance, bleeding-edge

The terms you will meet in 2026 and beyond — training pipeline, serving stack, agent security, and regulation.

Frontier model — A state-of-the-art model at the leading edge of capability; GPQA is designed so its difficulty holds even for "frontier AI systems," enabling scalable-oversight research. (GPQA (arXiv))

Post-training — Training done after pre-training to align/steer a model, such as RLHF or DPO, which fine-tune a pretrained model to human preferences. (DPO (arXiv))

Pretraining — Large-scale training on general-domain data that produces a base model later adapted to specific tasks. (LoRA (arXiv))

Distillation — Compressing the knowledge of a large model or ensemble into a single smaller model that is much easier to deploy, by training the small model to reproduce the larger one's outputs. (Knowledge Distillation (arXiv))

Synthetic data — Model-generated training data; Constitutional AI generates self-critiques and revisions and uses AI-generated preference labels (RLAIF) to train with far fewer human labels. (Constitutional AI (arXiv))

Reward model — A model fit to human preference labels that reflects those preferences and is then used as the reward signal in RLHF fine-tuning. (DPO (arXiv))

Preference tuning — Fine-tuning a model to align with human (or AI) preference judgments over pairs of outputs, as in RLHF, RLAIF, and DPO. (DPO (arXiv)), (Constitutional AI (arXiv))

DPO (Direct Preference Optimization) — A stable, lightweight alternative to RLHF that reparameterizes the reward model so the optimal policy can be extracted in closed form, solving preference alignment with a simple classification loss and no RL sampling. (DPO (arXiv))

Constitutional AI — Training a harmless assistant via self-improvement guided only by a list of rules/principles, using self-critique/revision (supervised) and RL from AI Feedback (RLAIF) instead of human harm labels. (Constitutional AI (arXiv))

Test-time compute — Spending more computation at inference to improve answers (e.g., generating reasoning steps); chain-of-thought prompting elicits intermediate reasoning that boosts complex-task accuracy. (Chain-of-Thought (arXiv))

Reasoning model — A model that produces intermediate reasoning; note that reasoning tokens, while not shown via the API, occupy context and are billed as output tokens. (OpenAI pricing)

Long-context model — A model supporting very large context windows (e.g., ~1.05M tokens for some OpenAI models), enabled in part by attention methods like FlashAttention that allow longer contexts. (OpenAI pricing), (FlashAttention (arXiv))

Multimodal model — A model handling multiple input types; OSWorld evaluates multimodal LLM/VLM-based agents that must ground actions in on-screen GUI images. (OSWorld (arXiv))

Vision-language model (VLM) — A model combining vision and language; OSWorld benchmarks "LLM/VLM-based agents" that struggle with GUI grounding. (OSWorld (arXiv))

Speech-to-text (ASR) — Automatic speech recognition; beam search is noted as "best suited for input-grounded tasks, like describing an image or speech recognition." (Hugging Face)

Text-to-speech (TTS) — Turning text into lifelike spoken audio via a speech-generation model (e.g., OpenAI's gpt-4o-mini-tts), with support for narration, multiple languages, and real-time streaming. (OpenAI docs)

Diffusion model — A class of latent-variable generative models (inspired by nonequilibrium thermodynamics) that produce high-quality samples, e.g., image synthesis, by learning to reverse

a gradual noising process. (Denoising Diffusion Probabilistic Models (arXiv))

Speculative decoding — An algorithm that speeds sampling from autoregressive models by computing several tokens in parallel — a faster model proposes tokens that the large model verifies — without changing the output distribution (2x-3x speedup on T5-XXL). (Speculative decoding (arXiv))

Draft model — The smaller/efficient approximation model whose outputs the large model runs on in parallel during speculative decoding to propose candidate tokens. (Speculative decoding (arXiv))

Verification model — The large target model that checks the draft's proposed tokens via speculative execution and a sampling method that leaves the distribution unchanged. (Speculative decoding (arXiv))

Prefix caching — Caching shared prompt prefixes so repeated computation is avoided; SGLang lists prefix caching (RadixAttention) among its low-latency features, and vLLM enables flexible KV-cache sharing across requests. (SGLang), (PagedAttention/vLLM (arXiv))

Continuous batching — Batching incoming requests continuously to increase total serving throughput. (Hugging Face (TGI))

Scheduler — The component managing how requests/memory are batched and allocated; PagedAttention manages KV memory in pages (OS-paging-inspired) to pack more requests. (PagedAttention/vLLM (arXiv))

Kernel fusion — **INFERENCE** — the term is standard GPU-engineering usage; no single canonical AI-specific source located. Combining several small GPU operations into one fused kernel to cut memory traffic and launch overhead; FlashAttention's tiling is a well-known example applied to attention. (FlashAttention (arXiv))

Tensor parallelism — A model-parallel method that distributes an individual layer's parameter tensor across GPUs, reducing model-state and activation memory per GPU. (NVIDIA)

Pipeline parallelism — Assigning consecutive layers/segments of a network to different GPUs so each processes different stages sequentially. (NVIDIA)

Expert parallelism — Distributing the experts of a Mixture-of-Experts model across GPUs, applied only to the expert layers without changing the parallel mapping of the rest. (NVIDIA)

Sharding — Splitting model tensors/state across devices; tensor parallelism partitions a layer's parameter tensor across GPUs to reduce per-GPU memory. (NVIDIA)

Checkpointing — In training generally, the practice of periodically saving model state so runs can be resumed or rolled back; PyTorch also uses the term for activation checkpointing (recomputing activations to save memory). (PyTorch)

Activation checkpointing — A memory-saving training technique that skips storing intermediate activations on the forward pass and recomputes them during the backward pass, trading extra compute for lower memory. (PyTorch)

Throughput optimization — Maximizing served volume per unit time via techniques like continuous batching, tensor parallelism, and PagedAttention's near-zero KV-cache waste. (Hugging Face (TGI)), (PagedAttention/vLLM (arXiv))

Latency optimization — Minimizing response delay; SGLang targets low-latency inference and speculative decoding accelerates generation without changing outputs. (SGLang), (Speculative decoding (arXiv))

Serving stack — The software that deploys and serves LLMs; TGI is a toolkit for deploying and serving LLMs, and recommends vLLM and SGLang as optimized inference engines going forward. (Hugging Face (TGI)) vLLM — An LLM serving system built on PagedAttention that achieves near-zero KV-cache waste and 2-4x higher throughput than prior systems at equal latency. (PagedAttention/vLLM (arXiv))

SGLang — A production-level inference framework designed for low-latency, high-throughput serving across single GPUs to distributed clusters, using RadixAttention, prefix caching, and multi-GPU parallelism.

TGI (Text Generation Inference) — A Hugging Face toolkit for deploying and serving popular open-source LLMs with continuous batching, tensor parallelism, and Flash/Paged attention (now in maintenance mode). (Hugging Face (TGI))

CUDA — NVIDIA's parallel computing platform and programming model for its GPUs. (NVIDIA)

ROCm — AMD's mostly open-source software stack that provides tools for programming AMD GPUs, from low-level kernels to high-level applications. (AMD)

Unified memory — A memory architecture in which CPU and GPU share the same system memory (e.g., Apple Silicon), which lets local LLM runtimes like llama.cpp load large models without a separate discrete VRAM pool. (Apple Developer), (llama.cpp)

Memory fragmentation — Wasted, non-contiguous free memory; inefficient KV-cache management wastes memory through fragmentation and duplication, which PagedAttention addresses via paging. (PagedAttention/vLLM (arXiv))

Out-of-memory (OOM) — Running out of device memory; QLoRA adds paged optimizers specifically to manage memory spikes during training, and hybrid CPU+GPU inference helps run models larger than VRAM. (QLoRA (arXiv)), (llama.cpp)

Guarded autonomy — **INFERENCE** — practitioner shorthand; no canonical source located. Letting an agent act on its own within explicit limits (allowed tools, allowed data, allowed budgets, required approvals for risky actions); the EU AI Act requires "appropriate human oversight" for high-risk systems, which is the regulatory anchor for this pattern. (EU AI Act)

Human-in-the-loop — Keeping human oversight over AI decisions; the EU AI Act requires "appropriate human oversight measures" for high-risk AI systems. (EU AI Act)

Policy enforcement — Enforcing rules/principles on model behavior; Constitutional AI enforces a "constitution" of principles to control behavior with far fewer human labels. (Constitutional AI (arXiv))

Sandbox — An isolated execution environment for agent actions; OSWorld provides a controlled real computer environment for execution-based evaluation across operating systems. (OSWorld (arXiv))

Isolation boundary — A separation preventing data/state leakage between tenants; Anthropic states cache entries are isolated between organizations and between workspaces within an organization. (Anthropic)

Blast radius — The set of assets, identities, data, and downstream systems that can be reached from a single compromised component; segmented isolation is used to keep it small. (Anthropic RSP)

Observability — Instrumenting a system to monitor behavior; TGI is production-ready with distributed tracing (OpenTelemetry) and Prometheus metrics. (Hugging Face (TGI))

Tracing — Recording the path of requests/operations for debugging and monitoring; TGI supports distributed tracing with OpenTelemetry. (Hugging Face (TGI))

Audit logging — Logging activity for traceability; the EU AI Act requires high-risk systems to log activity "to ensure traceability of results." (EU AI Act)

Model governance — Managing model risk and documentation across the lifecycle; the NIST AI RMF helps incorporate trustworthiness into AI design, development, use, and evaluation, and model cards document intended use and performance. (NIST AI RMF), (Model Cards (arXiv))

Data governance — Managing training/operational data quality and risk; the EU AI Act requires high-quality datasets to minimize discriminatory outcomes, and OWASP flags data-integrity risks in the ML supply chain. (EU AI Act), (OWASP)

AI safety case — **INFERENCE** — emerging governance term; no single canonical source located. A structured written argument, with evidence, that a specific AI system is acceptably safe to deploy in a specific context; borrowed from safety-case practice in aviation and nuclear, and increasingly used alongside model/system cards. (OpenAI)

Model card — A short document accompanying a trained model that reports benchmarked performance across conditions/groups and discloses intended use and evaluation procedures. (Model Cards (arXiv))

System card — An end-to-end safety assessment of a deployed model, covering capabilities, limitations, safety evaluations, and mitigations for current and frontier risks (e.g., OpenAI's GPT-4o system card). (OpenAI)

General-purpose AI (GPAI) — Models that can perform a wide range of tasks and underpin many downstream AI systems; under the EU AI Act, providers face transparency and copyright rules, with extra duties for models posing systemic risk. (EU AI Act)

EU AI Act — The first comprehensive legal framework on AI (Regulation (EU) 2024/1689), using a risk-based approach with four risk levels to foster trustworthy AI; it entered into force on 1 August 2024. (EU AI Act)

NIST AI RMF — The NIST AI Risk Management Framework, a voluntary framework to better manage AI risks to individuals, organizations, and society and to build trustworthiness into AI design, development, use, and evaluation. (NIST AI RMF)

Secure AI deployment — Deploying AI with controls against AI-specific threats; MITRE ATLAS catalogs adversary tactics/techniques against AI systems, and OWASP enumerates top LLM risks such as prompt injection and supply-chain vulnerabilities. (MITRE ATLAS), (OWASP)

Agentic workflow security — Securing tool-using agent workflows; OWASP notes RAG and fine-tuning do not fully mitigate prompt injection, which can force models to misuse data or violate guidelines. (OWASP)

Prompt injection — A vulnerability where user prompts alter an LLM's behavior or output in unintended ways, even via inputs imperceptible to humans, potentially bypassing safety measures. (OWASP)

Tool poisoning — A supply-chain-style manipulation where external components are tampered with or poisoned; OWASP notes third-party models and data can be manipulated through tampering or poisoning attacks. (OWASP)

Data exfiltration risk — The risk that an LLM discloses PII, proprietary algorithms, or confidential data through its output, causing unauthorized data access and IP breaches. (OWASP)

Supply chain risk for models — Vulnerabilities affecting the integrity of training data, models, and deployment platforms — including third-party pre-trained models and malicious LoRA adapters — that can cause biased outputs, breaches, or failures. (OWASP)

Model provenance — Assurance about where a model came from; OWASP notes there are currently "no strong provenance assurances in published models," and model cards offer no guarantee of a model's origin. (OWASP)

Watermarking — Embedding signals into generated text that are invisible to humans but algorithmically detectable from a short token span (e.g., softly promoting a randomized "green" token set during sampling). (Watermarking for LLMs (arXiv))

Evaluation harness — The framework that runs models against benchmark tasks with reproducible scoring; SWE-bench and OSWorld provide execution-based evaluation setups. (SWE-bench (arXiv)), (OSWorld (arXiv))

Regression testing for models — **INFERENCE** — extends standard software-testing usage. Re-running a fixed evaluation set (e.g., an execution-based benchmark like SWE-bench, plus internal task suites) after every prompt, model, or scaffold change to detect quality regressions before rollout. (SWE-bench (arXiv))

Canary rollout — A progressive deployment strategy that splits traffic between the current and new version and rolls out to a small subset of users before going 100%, so issues surface with limited exposure. (Google Cloud)

Rollback strategy — **INFERENCE** — standard deployment concept applied to models. A pre-planned way to revert to the previous known-good model, prompt, or agent scaffold if a canary or production release regresses; typically paired with canary rollout. (Google Cloud)

Autonomous agent risk — The risk from agents acting with minimal human intervention; the EU AI Act requires human-oversight measures for high-risk systems, and OSWorld shows current agents still fail most real computer tasks. (EU AI Act), (OSWorld (arXiv))

Inter-agent trust — Trust between communicating agents; A2A is designed so agents can "securely exchange information" and coordinate actions across platforms. (Google Developers Blog)

Capability boundary — **INFERENCE** — practitioner term; no single canonical source located. The line between what an agent is allowed and equipped to do versus what it is not; the EU AI Act's risk tiers formalize a related idea by imposing stricter obligations on higher-risk use cases. (EU AI Act)

Delegation policy — Rules governing how work is handed off between agents; in the orchestrator-workers pattern a central LLM "delegates" subtasks to worker LLMs and synthesizes results. (Anthropic)

Machine identity for agents — A non-human identity — credentials, permissions, and lifecycle controls — assigned to a software actor (workload, service, or agent) that authenticates and acts without a human initiating each request; NIST is standardizing management and protection of such identities. (NIST)

Protocol interoperability — The ability of agents built on a shared protocol to work together; A2A lets developers build agents that connect with any other A2A agent and combine agents from various providers, complementing MCP. (Google Developers Blog)

Emerging agent standards — New open protocols standardizing agent connectivity — MCP (app-to-tool context), A2A (agent-to-agent), and ANP (an "HTTP of the Agentic Web"). (Model Context Protocol), (Google Developers Blog), (Agent Network Protocol)

▪▪▪ SECTION 11

Interpretability, oversight, and adversarial evaluation

How to interpret what a model just did, keep humans in the decision loop, and probe the system adversarially. Every term in this section maps to a NIST AI RMF function (GOVERN, MAP, MEASURE, MANAGE) and either a canonical library, a standards paper, or a named community source.

SHAP (SHapley Additive exPlanations) — A game-theoretic feature-attribution method that explains any ML model by assigning each feature a Shapley value; it unifies additive-attribution methods and satisfies local accuracy, missingness, and consistency. Maps to AI RMF MEASURE. (SHAP docs), (SHAP (Nature MI))

Shapley value — The average marginal contribution of a feature across all coalitions of the other features, weighted so contributions sum fairly to the prediction; in SHAP this is the per-feature attribution for one instance. (Molnar IML)

Marginal contribution — The change in model output when one feature is added to a coalition of already-present features; SHAP averages these across coalitions to compute a Shapley value. (Molnar IML)

Tree SHAP — An exact SHAP variant for tree ensembles (XGBoost, LightGBM, random forests) that computes Shapley values in polynomial time and is the default choice for tabular tree models. (SHAP tree (arXiv))

Kernel SHAP — A model-agnostic SHAP variant that samples feature coalitions, weights them with the SHAP kernel, and fits a weighted linear model; connects SHAP to LIME conceptually. (SHAP docs), (Molnar IML)

Background data — The reference dataset SHAP uses to represent "absent" features when computing attributions; changing the background changes the meaning of the explanation, which is a governance-relevant choice. (SHAP docs)

Local explanation — A per-instance attribution vector that says why the model produced one specific prediction; SHAP and LIME are the two most-used local explainers. (LIME (arXiv)), (SHAP docs)

Global explanation — Model behavior aggregated across many predictions — mean absolute SHAP for feature importance, dependence plots, and interaction plots; Tree SHAP makes global aggregation of Shapley values tractable. (SHAP (Nature MI))

Additive feature attribution — The class of explainers that expresses an explanation as a linear sum of feature contributions: $g(z') = \varphi_0 + \Sigma\, \varphi_j\, z_j'$. Lundberg and Lee proved only Shapley values satisfy all three additive-attribution axioms. (SHAP docs)

LIME (Local Interpretable Model-agnostic Explanations) — A local surrogate method that explains one prediction by perturbing the instance, querying the black-box model, weighting by proximity, and fitting a simple interpretable model to the neighborhood. (LIME (arXiv)), (marcotcr/lime)

Select instance — Step 1 of LIME: pick the specific record, prompt, image, or text example whose prediction needs explanation. (LIME (arXiv))

Perturb data — Step 2 of LIME: generate variations of the selected instance (masked features, super-pixel toggles, token drops) that will probe the model's local behavior. (LIME (arXiv)), (marcotcr/lime)

Get predictions — Step 3 of LIME: run the perturbed samples through the black-box model and collect its outputs; those become training data for the local surrogate. (LIME (arXiv))

Weight proximity — Step 4 of LIME: assign each perturbed sample a weight based on how close it is to the original instance, using a proximity kernel; this defines the neighborhood. (LIME (arXiv)), (Molnar IML)

Fit local surrogate — Step 5 of LIME: fit a sparse, interpretable model (usually lasso linear regression or a short decision tree) on the weighted perturbed data; the surrogate's coefficients are the explanation. (LIME (arXiv))

Kernel width — The bandwidth of LIME's proximity kernel; wider kernels blur the neighborhood, narrower kernels make LIME unstable — a documented failure mode of the method. (Molnar IML)

Local fidelity — How faithfully the LIME surrogate matches the black-box model in the neighborhood of the selected instance; LIME optimizes local fidelity, not global accuracy. (LIME (arXiv))

Sparse explanation — A LIME explanation constrained to a small fixed number of features so a human can read it; sparsity is a design choice that trades detail for interpretability. (LIME (arXiv))

LIME instability — The empirical fact that LIME explanations vary across runs on the same input due to perturbation randomness and kernel-width sensitivity; a widely cited weakness relative to SHAP. (Molnar IML), (Fooling LIME/SHAP (arXiv))

Surrogate model — A simpler, interpretable model trained to approximate a more complex model — either locally (LIME) or globally (a decision tree fit to a neural network's outputs). (Molnar IML)

Global surrogate — An interpretable model (linear regression, decision tree) fit to a black-box model's predictions across the whole dataset to approximate it; useful when a single global explanation is required. (Molnar IML)

Model-agnostic explanation — An explainer that works on any model because it treats the model as a black box that maps inputs to outputs; SHAP's Kernel variant and LIME are the canonical examples. (Molnar IML)

Manipulability of explanations — The demonstrated fact that biased models can be engineered to produce innocuous LIME/SHAP explanations by exploiting the out-of-distribution nature of perturbed samples — an active attack surface. (Fooling LIME/SHAP (arXiv))

Human review — Structured examination of AI outputs, decisions, or risk signals by a person before those outputs are relied upon in a sensitive context; the operational form of NIST AI RMF human-centered oversight. Maps to AI RMF GOVERN. (NIST AI 100-1)

Human oversight — The governance function that keeps humans empowered, accountable, and able to intervene in AI lifecycle decisions; the EU AI Act requires "appropriate human oversight measures" for high-risk systems. (NIST AI 100-1), (EU AI Act)

Human-in-the-loop control point — A specific step in an agentic workflow where the system pauses for explicit human approval before taking a high-risk action (e.g., sending money, making a decision about a person); OWASP recommends this pattern for privileged operations. (OWASP LLM01)

Accountability — The assignment of responsibility for mapping, measuring, managing, and responding to AI risks across the lifecycle; NIST AI RMF treats accountability structures as a GOVERN-function requirement. (NIST AI 100-1)

Transparency — The degree to which information about an AI system, its use, and its risk posture is communicated to relevant stakeholders; a NIST trustworthy AI characteristic. (NIST AI 100-1)

Explainability (RMF) — The trustworthy-AI characteristic requiring that system outputs and behavior be meaningfully explainable to users, experts, and other stakeholders; operationalized by SHAP, LIME, and counterfactuals. Maps to AI RMF MEASURE. (NIST AI 100-1), (Molnar IML)

Interpretability (RMF) — The degree to which a human can understand the basis of an AI output or model behavior; paired with explainability as a NIST trustworthy characteristic. (NIST AI 100-1)

Adversarial testing — Deliberate probing of an AI system to discover failure modes, abuse paths, and unsafe behavior under hostile or edge-case conditions; OWASP frames it across four surfaces (model, implementation, infrastructure, runtime). Maps to AI RMF MEASURE. (OWASP GenAI RT)

AI red teaming — The broader security-and-assurance practice that simulates realistic attacks and abuse to evaluate model, application, and operational resilience; more than jailbreak testing, includes threat models and evaluation rigor. (OWASP GenAI RT)

Model evaluation (RT) — The red-team activity of testing the model itself for safety, failure modes, bias, and exploitable behavior; distinct from implementation testing that targets the app layer. (OWASP GenAI RT)

Implementation testing — The red-team activity of probing the application layer, orchestration, prompt handling, wrappers, and integration logic around the model. (OWASP GenAI RT)

Infrastructure assessment — The red-team activity of probing the environment hosting and serving the model — deployment, access paths, and supporting systems. (OWASP GenAI RT)

Runtime behavior analysis — Examination of what the AI system actually does during operation under normal and adversarial conditions; one of OWASP's four core red-team activity areas. (OWASP GenAI RT)

Evaluation rigor — The degree to which adversarial testing is systematic, repeatable, comprehensive, and evidence-based; OWASP distinguishes rigorous testing from superficial jailbreak-only offerings. (OWASP GenAI RT)

Adversarial ML taxonomy — NIST's standardized vocabulary of AML attack classes — evasion, poisoning, privacy, and abuse — and mitigations; the reference terminology auditors quote in reports. (NIST AI 100-2 E2025)

Jailbreak — A prompt or prompt sequence that causes an LLM to bypass its safety training; DEF CON's Generative Red Team (GRT) exercises documented that role-play, story-writing, and context switching succeed most often. (AI Village GRT), (Humane Intelligence 2024)

Generative Red Team (GRT) — The DEF CON AI Village exercise series (GRT, GRT2, GRT3) run with Humane Intelligence; the largest public adversarial-testing dataset for generative AI systems. (AI Village GRT), (Humane Intelligence 2024)

MITRE ATLAS (workflow) — The ATT&CK-style knowledge base of tactics, techniques, and case studies specifically targeting AI systems across the lifecycle; the practical starting point for AI-system threat modeling. (MITRE ATLAS), (MITRE ATLAS (GitHub))

AI RMF functions — The four core functions of NIST AI RMF 1.0: GOVERN (culture and policy), MAP (context and scope), MEASURE (metrics and evaluation), and MANAGE (prioritize and respond). Every term in this section maps to one or more. (NIST AI 100-1)

▪▪▪ FIELD NOTES

What matters operationally

Fourteen takeaways to keep after everything else fades. Each is defensible against the linked source.

[01] Own the token math. Cost and latency both scale with tokens in and tokens out. Long prompts, long context, and reasoning tokens all bill as output. Cached input is roughly an order of magnitude cheaper on some providers. (OpenAI pricing), (Anthropic)

[02] KV cache is the real memory budget on inference. Weights dominate at load, then the per-request KV cache grows with sequence length and batch size. PagedAttention/vLLM is the standard fix. (PagedAttention/vLLM (arXiv))

[03] Quantization is the cheapest local win. GGUF plus Q4/Q5/Q8 in llama.cpp, Ollama, or LM Studio is how teams run 70B-class models on workstations. INT4 gives large speedups with minimal accuracy loss. (llama.cpp), (NVIDIA)

[04] Prefer RAG plus reranking over blind long-context prompts. Hybrid retrieval (BM25 + embeddings) with a cross-encoder reranker keeps context small, fast, and grounded. Anthropic's contextual retrieval cut top-20 failures by 35%. (Anthropic)

[05] Agents are ReAct loops with tools, not magic. Effective agents interleave reasoning and tool calls, then use environment feedback as ground truth. That framing tells you exactly where to add logging, retries, and human review. (ReAct (arXiv)), (Anthropic)

[06] MCP, A2A, and ANP are the interop bets to track. MCP standardizes app-to-tool context; A2A standardizes agent-to-agent; ANP aims to be the HTTP of the agentic web. Pick stacks that speak these to avoid lock-in. (Model Context Protocol), (Google Developers Blog), (Agent Network Protocol)

[07] Assume prompt injection until proven otherwise. OWASP's LLM Top 10 lists prompt injection as risk #1. RAG and fine-tuning do not fully mitigate it. Sandbox tool actions and constrain what an agent can reach. (OWASP), (OWASP LLM01)

[08] Treat the model supply chain like any other supply chain. Third-party weights, LoRA adapters, and datasets can carry hidden bias or backdoors; there are no strong provenance assurances today. Track model card, system card, and hash of what you deployed. (OWASP), (Model Cards (arXiv))

[09] Ship behind evals and canaries, not vibes. Every prompt, model, or scaffold change should re-run a fixed evaluation harness. Roll out with a canary and a documented rollback so a regression is contained. (SWE-bench (arXiv)), (Google Cloud)

[10] Governance is a shipping requirement now. The EU AI Act (Regulation (EU) 2024/1689) mandates risk assessment, logging, human oversight, and robustness for high-risk systems, plus duties on general-purpose AI providers. The NIST AI RMF gives you the voluntary framework to satisfy those obligations. (EU AI Act), (NIST AI RMF)

[11] Explanations are evidence, not truth. SHAP and LIME both make defensible per-prediction attributions, but SHAP is consistent by construction and LIME is not — audit reports should prefer SHAP for regulated decisions and always disclose the background dataset or kernel width used. (SHAP (Nature MI), Molnar IML)

[12] Assume the explanation itself can be attacked. Slack et al. showed biased models can be engineered to produce clean LIME and SHAP outputs; when the stakes are high, treat the explanation as one signal among several, not proof. (Fooling LIME/SHAP (arXiv))

[13] Wire adversarial testing to the four OWASP surfaces. Cover model, implementation, infrastructure, and runtime behavior — jailbreak-only testing is not enough for the GenAI Red Teaming Guide. (OWASP GenAI RT)

[14] Use NIST AI 100-2 E2025 as your shared vocabulary. When engineers, auditors, and executives argue about "model attacks," pin the discussion to the taxonomy (evasion, poisoning, privacy, abuse) so the fix maps to a documented class. (NIST AI 100-2 E2025)

■■■ SOURCE INDEX

Every source cited in this glossary

Deduplicated by URL; inline citations throughout the document link directly to these references. Entries added in v2026.08 are marked **[new]**.

Agent Network Protocol — https://github.com/agent-network-protocol/AgentNetworkProtocol

AI Village GRT **[new]** — https://aivillage.org/defcon%2031/generative-recap/

AMD — https://rocm.docs.amd.com/en/latest/what-is-rocm.html

Anthropic — https://www.anthropic.com/news/contextual-retrieval

Anthropic RSP — https://www.anthropic.com/responsible-scaling-policy

Apple Developer — https://developer.apple.com/documentation/metal/choosing-a-resource-storage-mode-for-apple-gpus

Attention Is All You Need (arXiv) — https://arxiv.org/abs/1706.03762

Chain-of-Thought (arXiv) — https://arxiv.org/abs/2201.11903

Chinchilla (arXiv) — https://arxiv.org/abs/2203.15556

Constitutional AI (arXiv) — https://arxiv.org/abs/2212.08073

Denoising Diffusion Probabilistic Models (arXiv) — https://arxiv.org/abs/2006.11239

Distillation (arXiv) — https://arxiv.org/abs/1503.02531

DPO (arXiv) — https://arxiv.org/abs/2305.18290

Epoch AI — https://epoch.ai/benchmarks/gpqa-diamond

EU AI Act — https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai

FlashAttention (arXiv) — https://arxiv.org/abs/2205.14135

Fooling LIME/SHAP (arXiv) **[new]** — https://arxiv.org/abs/1911.02508

Google Cloud — https://cloud.google.com/deploy/docs/deployment-strategies/canary

Google Developers Blog — https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/

GPQA (arXiv) — https://arxiv.org/abs/2311.12022

HackTricks AI **[new]** — https://hacktricks-wiki-hacktricks.mintlify.app/ai-security/overview

Hugging Face — https://huggingface.co/docs/transformers/en/glossary

Hugging Face (TGI) — https://huggingface.co/docs/text-generation-inference/en/index

Humane Intelligence 2024 **[new]** — https://humane-intelligence.org/wp-content/uploads/2025/09/2024-GenerativeAI-RedTeaming-TransparencyReport.pdf

IBM — https://www.ibm.com/think/topics/artificial-intelligence

InstructGPT (arXiv) — https://arxiv.org/abs/2203.02155

Length-Extrapolatable Transformer (arXiv) — https://arxiv.org/abs/2212.10554

LIME (arXiv) **[new]** — https://arxiv.org/abs/1602.04938

llama.cpp — https://github.com/ggml-org/llama.cpp

LM Studio — https://lmstudio.ai/docs/app

LoRA (arXiv) — https://arxiv.org/abs/2106.09685

marcotcr/lime **[new]** — https://github.com/marcotcr/lime

Mistral 7B (arXiv) — https://arxiv.org/abs/2310.06825

MITRE ATLAS — https://atlas.mitre.org/

MITRE ATLAS (GitHub) **[new]** — https://github.com/mitre-atlas/atlas-data

MMLU (arXiv) — https://arxiv.org/abs/2009.03300

Model Cards (arXiv) — https://arxiv.org/abs/1810.03993

Model Context Protocol — https://modelcontextprotocol.io/docs/getting-started/intro

Molnar IML **[new]** — https://christophm.github.io/interpretable-ml-book/

Multi-Query Attention (arXiv) — https://arxiv.org/abs/1911.02150

NIST — https://csrc.nist.rip/external/nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8320C.ipd.pdf
NIST AI 100-1 **[new]** — https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf
NIST AI 100-2 E2025 **[new]** — https://csrc.nist.gov/pubs/ai/100/2/e2025/final
NIST AI RMF — https://www.nist.gov/itl/ai-risk-management-framework
NVIDIA — https://docs.nvidia.com/nemo-framework/user-guide/latest/nemotoolkit/features/parallelisms.html
Ollama — https://github.com/ollama/ollama/blob/main/README.md
OpenAI — https://openai.com/index/gpt-4o-system-card/
OpenAI docs — https://platform.openai.com/docs/guides/function-calling
OpenAI pricing — https://platform.openai.com/docs/pricing
OSWorld (arXiv) — https://arxiv.org/abs/2404.07972
OWASP — https://genai.owasp.org/llmrisk/llm022025-sensitive-information-disclosure/
OWASP GenAI RT **[new]** — https://genai.owasp.org/resource/genai-red-teaming-guide/
OWASP LLM01 **[new]** — https://genai.owasp.org/llmrisk/llm01-prompt-injection/
PagedAttention/vLLM (arXiv) — https://arxiv.org/abs/2309.06180
Pinecone — https://docs.pinecone.io/guides/index-data/indexing-overview
PyTorch — https://pytorch.org/docs/stable/checkpoint.html
QLoRA (arXiv) — https://arxiv.org/abs/2305.14314
RAG (arXiv) — https://arxiv.org/abs/2005.11401
ReAct (arXiv) — https://arxiv.org/abs/2210.03629
RMSNorm (arXiv) — https://arxiv.org/abs/1910.07467
RoFormer/RoPE (arXiv) — https://arxiv.org/abs/2104.09864
Sentence-Transformers — https://www.sbert.net/examples/applications/cross-encoder/README.html
SGLang — https://docs.sglang.ai/
SHAP (Nature MI) **[new]** — https://www.nature.com/articles/s42256-019-0138-9
SHAP docs **[new]** — https://shap.readthedocs.io/
SHAP tree (arXiv) **[new]** — https://arxiv.org/abs/1802.03888
Speculative decoding (arXiv) — https://arxiv.org/abs/2211.17192
SWE-bench (arXiv) — https://arxiv.org/abs/2310.06770
SWE-Bench Pro (arXiv) — https://arxiv.org/abs/2509.16941
Switch Transformer (arXiv) — https://arxiv.org/abs/2101.03961
Watermarking for LLMs (arXiv) — https://arxiv.org/abs/2301.10226

▪▪▪ JOIN US

Home for Ethical Hackers

getblackhack.com · blackhackacademy.com · store.getblackhack.com

Nonprofit status pending. Contributions to BlackHack Society, Inc. are not currently tax-deductible. Sponsorships and services are B2B transactions.

© 2026 BlackHack Society. All rights reserved.

▪▪▪ ADDENDUM

Visual explainers: charts, graphs, and workflows

Twenty-three figures that turn the glossary's most load-bearing concepts into pictures. A01–A17 are the v2026.07 originals (preserved verbatim). A18–A23 are new in v2026.09 and unpack Section 11 — interpretability, oversight, and adversarial evaluation. Each caption points back to the section it supports and to the primary sources.

FIG A01 · **Map of AI** — pairs with Section 1 · Core AI and ML foundations. Sources: IBM

FIG A02 · **The LLM lifecycle** — pairs with Sections 2 and 10 · LLM basics; post-training. Sources: InstructGPT (arXiv), LoRA (arXiv)

FIG A03 · **Inside a decoder-only Transformer** — pairs with Section 3 · Transformer and model architecture. Sources: Attention Is All You Need (arXiv), FlashAttention (arXiv)

FIG A04 · **How a model picks the next token** — pairs with Sections 2–3 · sampling and decoding. Sources: Hugging Face

FIG A05 · **RAG pipeline** — pairs with Section 5 · Retrieval, search, and knowledge grounding. Sources: RAG (arXiv), Anthropic

FIG A06 · **The agent loop (ReAct)** — pairs with Section 6 · Agent and tool-use. Sources: ReAct (arXiv), Chain-of-Thought (arXiv)

FIG A07 · **Multi-agent topology and protocols** — pairs with Sections 6–7 · agents; protocols and ecosystem. Sources: Model Context Protocol, Google Developers Blog, Agent Network Protocol

FIG A08 · **LLM attack surface and defenses** — pairs with Section 10 · governance and security. Sources: OWASP, MITRE ATLAS

FIG A09 · **Ship behind evals and canaries** — pairs with Sections 8–9 and Field Note 09. Sources: SWE-bench (arXiv), Google Cloud

FIG A10 · **Local vs cloud decision flow** — pairs with Section 4 · Local LLM and hardware. Sources: Ollama, OpenAI pricing

FIG A11 · **LoRA and QLoRA fine-tuning economics** — pairs with Section 10 · post-training and optimization. Sources: LoRA (arXiv), QLoRA (arXiv)

FIG A12 · **Quantization memory ladder** — pairs with Section 4 · Local LLM and hardware. Sources: QLoRA (arXiv), llama.cpp

FIG A13 · **Why long context costs memory** — pairs with Sections 3–4 · KV cache and serving. Sources: PagedAttention/vLLM (arXiv), FlashAttention (arXiv)

FIG A14 · **Chinchilla scaling** — pairs with Section 10 · frontier and scaling laws. Sources: Chinchilla (arXiv)

FIG A15 · **Anatomy of API cost** — pairs with Section 9 and Field Note 01 · cost framing. Sources: OpenAI pricing

FIG A16 · **The evidence board** — pairs with Cross-cutting · headline numbers with primary sources. Sources: multiple

FIG A17 · **Benchmark landscape and the agent reality check** — pairs with Section 8 · Evaluation and benchmarks. Sources: SWE-bench (arXiv), SWE-Bench Pro (arXiv), GPQA (arXiv), OSWorld (arXiv), Epoch AI

FIG A18 · **Adversarial testing four surfaces** — pairs with Section 11 (new) · red-team scope. Sources: OWASP GenAI RT

FIG A19 · **SHAP vs LIME explanation flow** — pairs with Section 11 (new) · interpretability workflows. Sources: SHAP docs, LIME (arXiv), Molnar IML

FIG A20 · **Human-in-the-loop control point** — pairs with Section 11 (new) · oversight and governance. Sources: NIST AI 100-1, EU AI Act, OWASP LLM01

FIG A21 · **AI RMF crosswalk** — pairs with Section 11 (new) · NIST AI RMF mapping. Sources: NIST AI 100-1

FIG A22 · **MITRE ATLAS tactics matrix** — pairs with Section 11 (new) · threat modeling for AI systems. Sources: MITRE ATLAS, MITRE ATLAS (GitHub)

FIG A23 · **NIST AI 100-2 taxonomy tree** — pairs with Section 11 (new) · shared vocabulary for adversarial ML. Sources: NIST AI 100-2 E2025

▪▪▪ ORIGINAL FIGURES A01–A17

From v2026.07 — preserved verbatim

These eight pages hold the original A01–A17 figure set. Each page carries two figures in the layout used by the v2026.07 addendum. Captions are inside each figure.

▪▪▪ ADDENDUM

Visual explainers: charts, graphs, and workflows

Seventeen figures that turn the glossary's most load-bearing concepts into pictures. Each caption points back to the section it supports and to the primary sources behind the numbers.

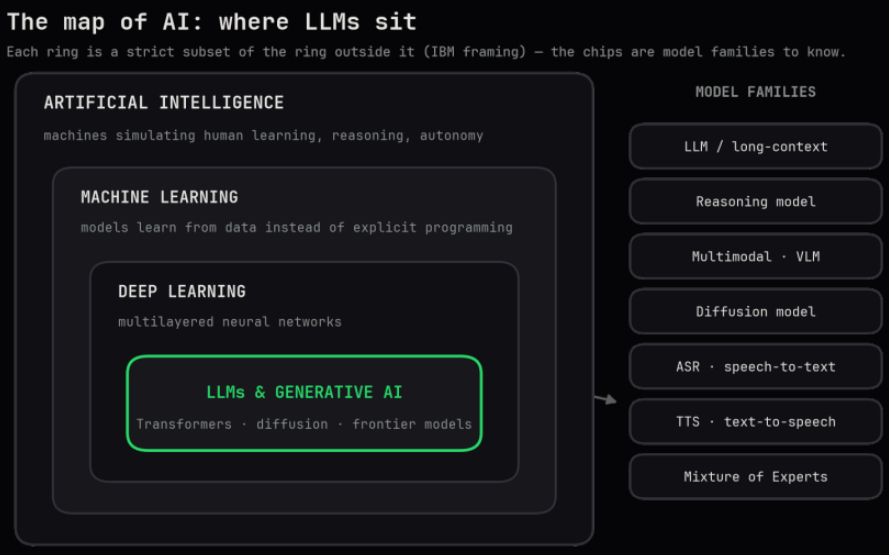


FIG A01 · Map of AI — pairs with Section 1 · Core AI and ML foundations. Sources: IBM

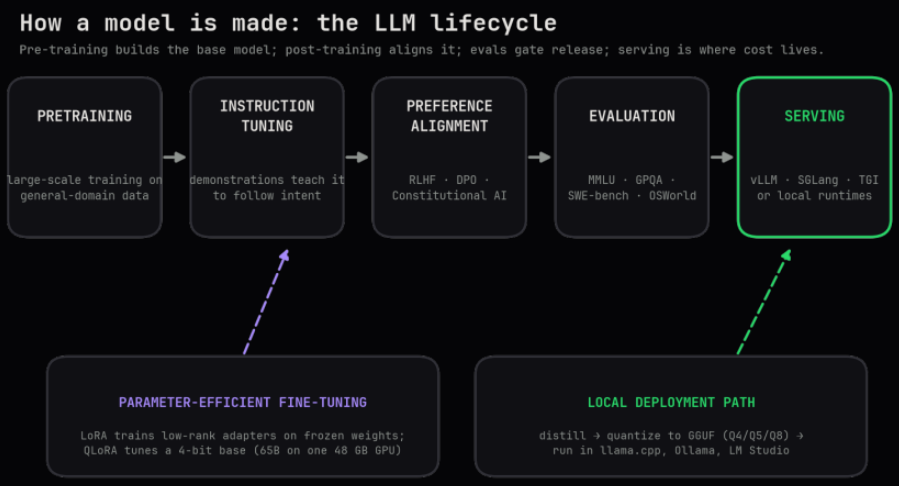


FIG A02 · The LLM lifecycle — pairs with Sections 2 and 10 · LLM basics; post-training. Sources: InstructGPT (arXiv) · LoRA (arXiv)

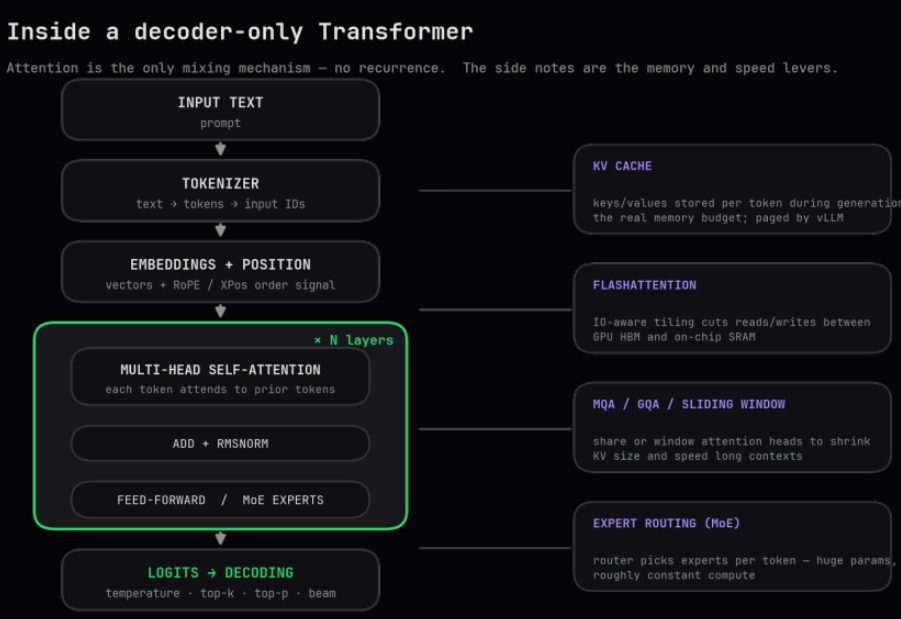


FIG A03 · Inside a decoder-only Transformer — pairs with Section 3 · Transformer and model architecture. Sources: Attention Is All You Need (arXiv) · FlashAttention (arXiv)

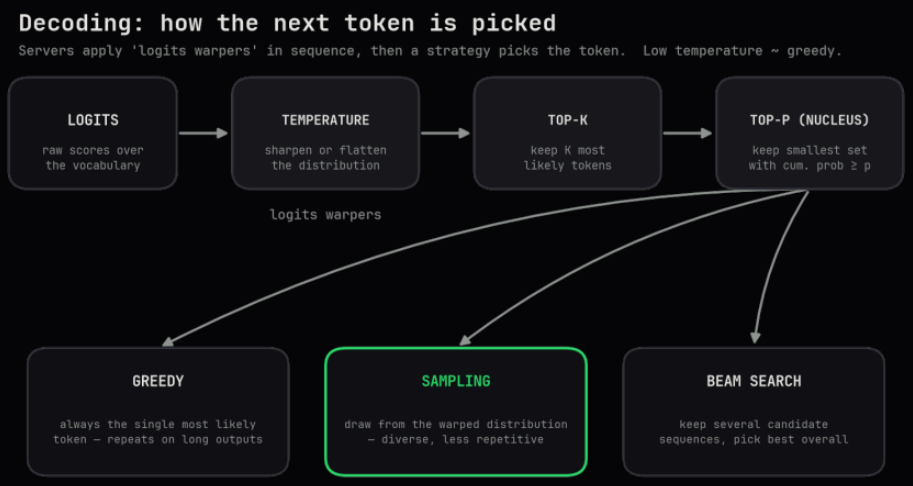


FIG A04 · How a model picks the next token — pairs with Sections 2-3 · sampling and decoding. Sources: Hugging Face

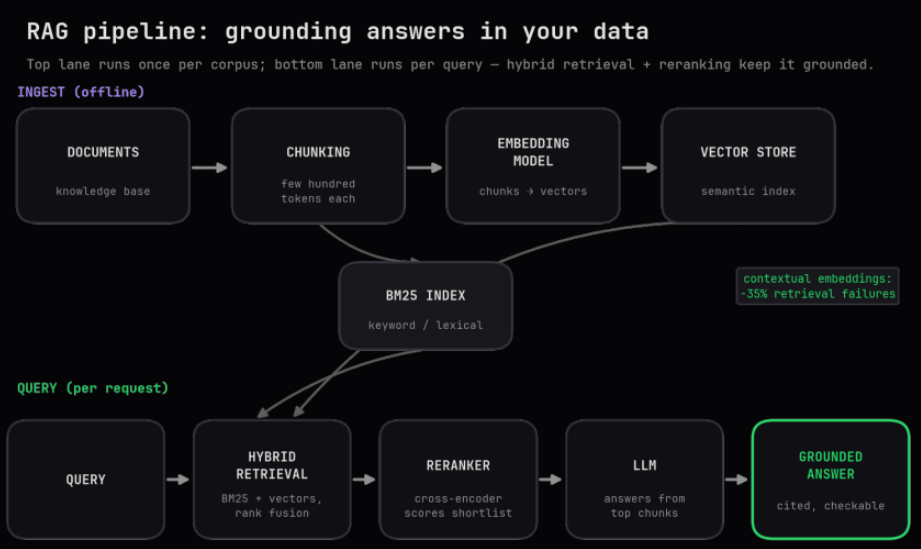


FIG A05 · RAG pipeline — pairs with Section 5 · Retrieval, search, and knowledge grounding. Sources: RAG (arXiv) · Anthropic



FIG A06 · The agent loop (ReAct) — pairs with Section 6 · Agent and tool-use. Sources: ReAct (arXiv) · Chain-of-Thought (arXiv)

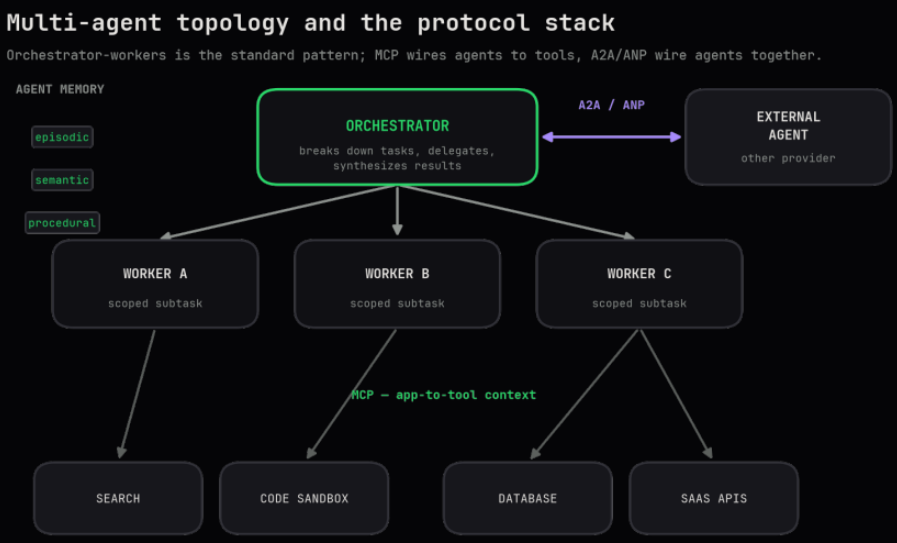


FIG A07 · Multi-agent topology and protocols — pairs with Sections 6-7 · agents; protocols and ecosystem. Sources: MCP · A2A



FIG A08 · LLM attack surface and defenses — pairs with Section 10 · governance and security. Sources: OWASP · MITRE ATLAS

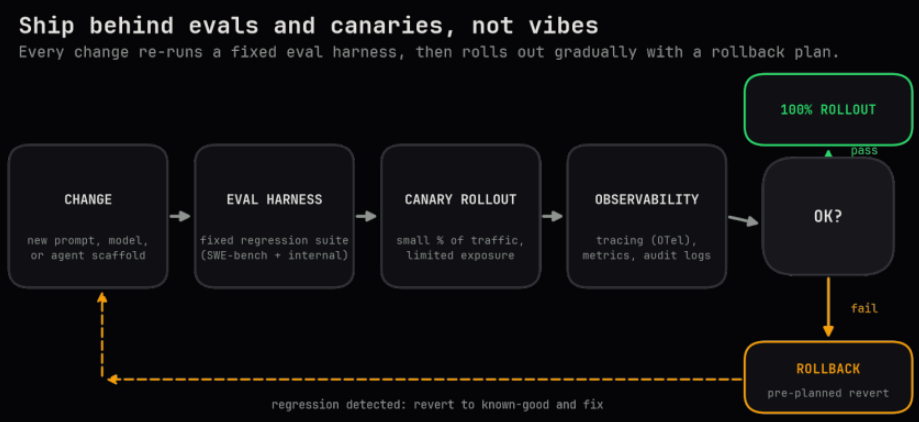


FIG A09 · Ship behind evals and canaries — pairs with Sections 8-9 and Field Note 09. Sources: SWE-bench (arXiv) · Google Cloud

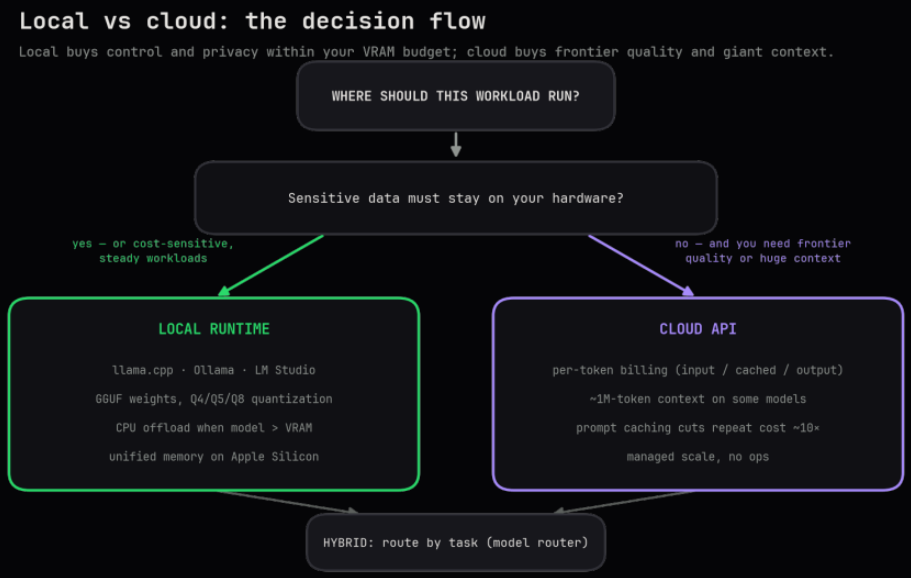


FIG A10 · Local vs cloud decision flow — pairs with Section 4 · Local LLM and hardware. Sources: Ollama · OpenAI pricing

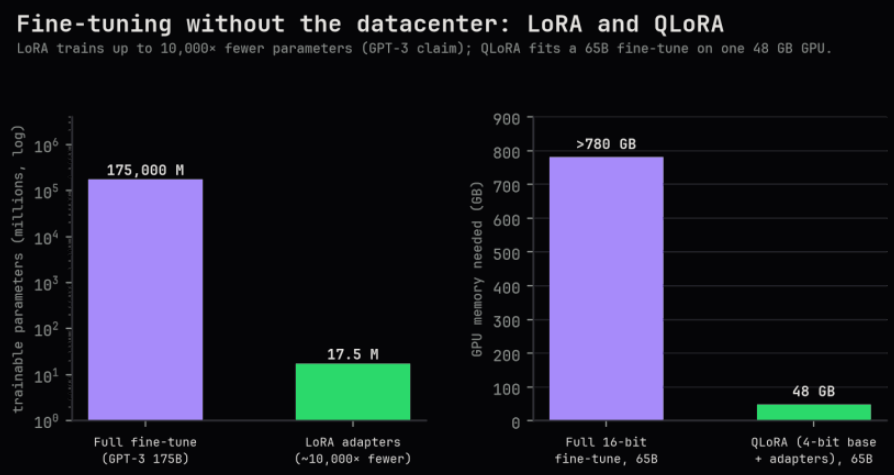


FIG A11 · LoRA and QLoRA fine-tuning economics — pairs with Section 10 · post-training and optimization. Sources: LoRA (arXiv) · QLoRA (arXiv)



FIG A12 · Quantization memory ladder — pairs with Section 4 · Local LLM and hardware. Sources: QLoRA (arXiv) · llama.cpp

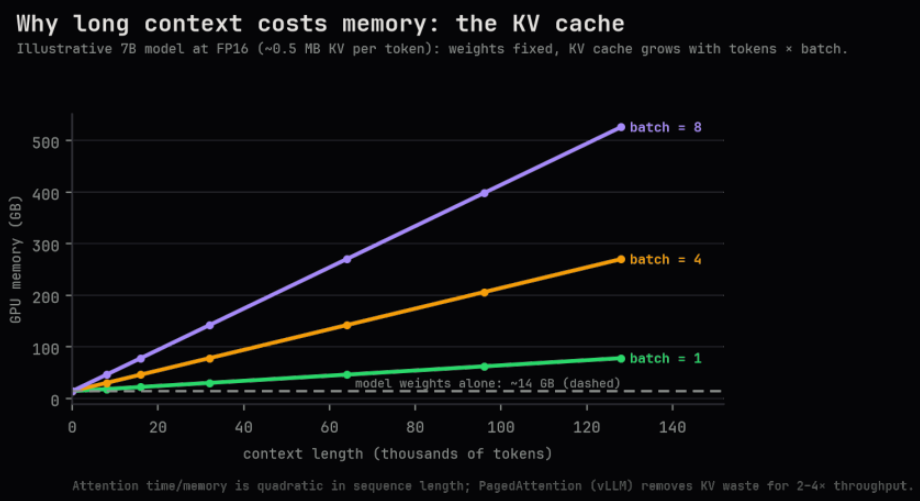


FIG A13 · Why long context costs memory — pairs with Sections 3-4 · KV cache and serving. Sources: PagedAttention/vLLM (arXiv) · FlashAttention (arXiv)

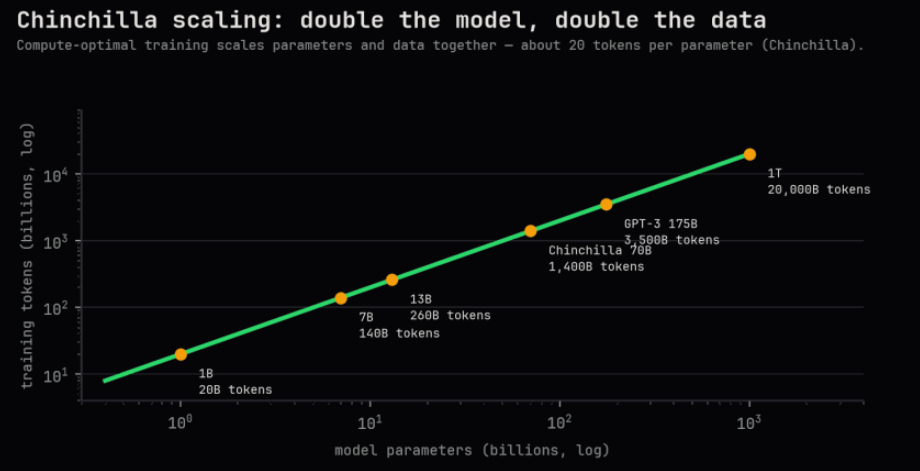


FIG A14 · Chinchilla scaling — pairs with Section 10 · frontier and scaling laws. Sources: Chinchilla (arXiv)



FIG A15 · Anatomy of API cost — pairs with Section 9 and Field Note 01 · cost framing. Sources: OpenAI pricing

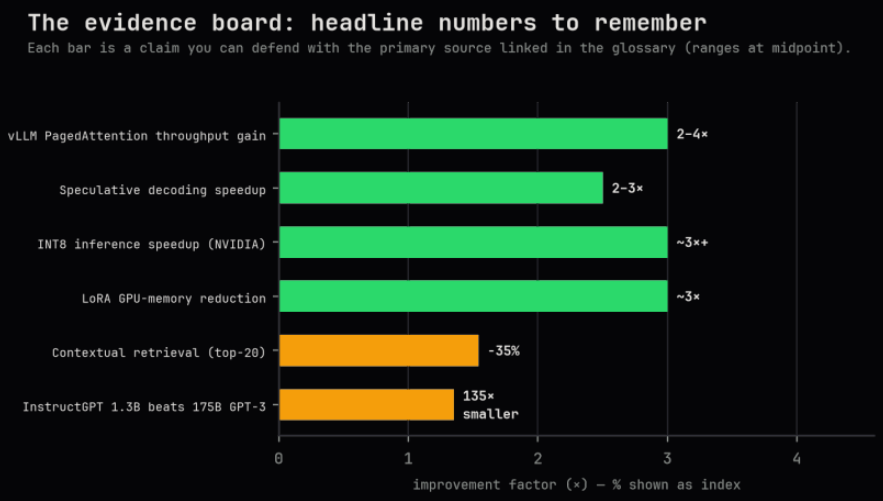


FIG A16 · The evidence board — pairs with Cross-cutting · headline numbers with primary sources. Sources: PagedAttention/vLLM (arXiv) · Speculative decoding (arXiv) · NVIDIA · LoRA (arXiv) · Anthropic · InstructGPT (arXiv)

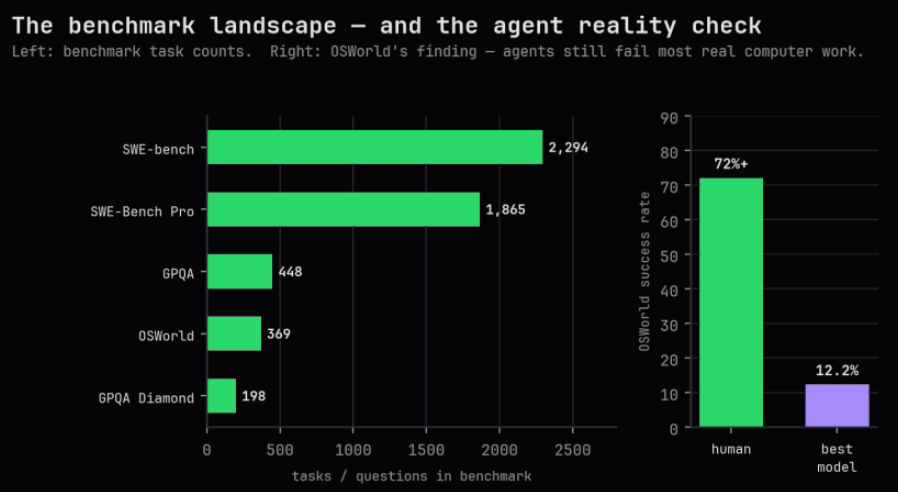


FIG A17 · Benchmark landscape and the agent reality check — pairs with Section 8 · Evaluation and benchmarks. Sources: SWE-bench (arXiv) · SWE-Bench Pro (arXiv) · GPQA (arXiv) · OSWorld (arXiv) · Epoch AI

▪▪▪ NEW FIGURES A18–A23

§11 explainers — new in v2026.09

Six figures that unpack the Section 11 vocabulary: adversarial testing scope, the two explanation methods, the human-review pause, the AI RMF crosswalk, MITRE ATLAS tactics, and the NIST AI 100-2 taxonomy tree.

Adversarial testing: the four OWASP surfaces

Rigorous GenAI red teaming covers all four — jailbreak-only testing is not enough.



FIG A18 · Adversarial testing four surfaces

pairs with §11 · Interpretability, oversight, and adversarial evaluation.

Sources: OWASP GenAI RT

SHAP vs LIME: two paths to a local explanation

LIME fits a local surrogate. SHAP allocates Shapley credit across coalitions.



SHAP: consistent, deterministic, expensive. LIME: sparse, fast, sensitive to kernel width.

FIG A19 · SHAP vs LIME explanation flow

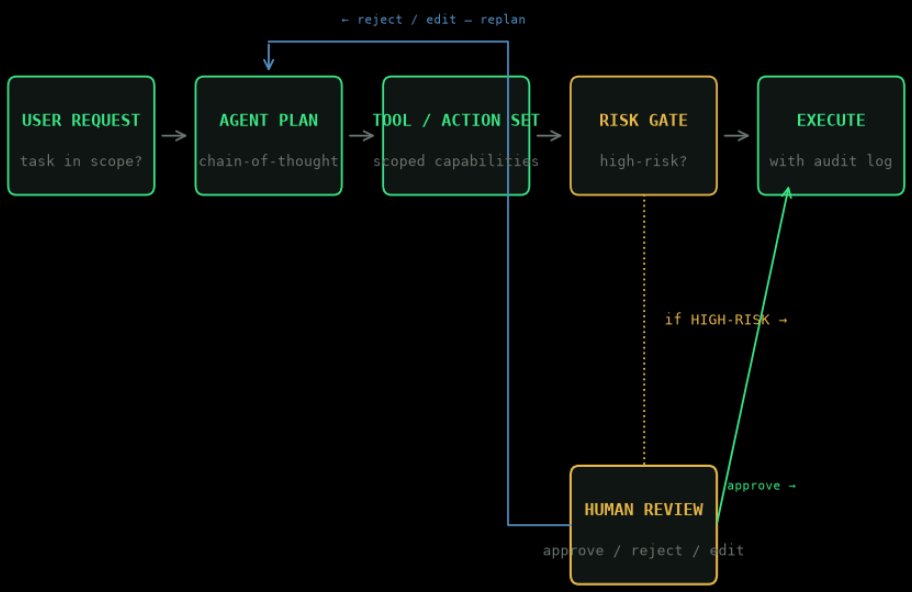
pairs with §11 · Interpretability workflows.

Sources: SHAP docs · LIME (arXiv) · Molnar IML

Human-in-the-loop: where a person interrupts an agent

Autonomous when scoped is fine. High-risk actions pause for explicit approval.

Regulatory anchor: EU AI Act 'appropriate human oversight' · NIST AI RMF GOVERN function



every branch writes an audit-log entry: (agent, tool, args, human-decision, timestamp)

FIG A20 · Human-in-the-loop control point

pairs with §11 · Oversight and governance.

Sources: NIST AI 100-1 · EU AI Act · OWASP LLM01

AI RMF crosswalk: what §11 terms map to which function

Every interpretability, oversight, and adversarial-testing term ties to GOVERN / MAP / MEASURE / MANAGE.

GOVERN	MAP	MEASURE	MANAGE
culture, policy, accountability	context, scope, impacts	metrics, evaluation, evidence	prioritize, respond, risk treatment
· Human review	· MITRE ATLAS	· SHAP	· HITL control point
· Human oversight	· Threat model	· LIME	· Rollback strategy
· Accountability	· Adversarial ML taxonomy	· Local / Global explanat…	· Canary rollout
· Transparency	· Model card	· Explainability (RMF)	· Manipulability of expl.
· AI safety case	· System card	· Interpretability (RMF)	· Supply chain risk
· Data governance	· Machine identity (agent…	· Adversarial testing	· Tool poisoning
· Model governance		· AI red teaming	· Autonomous agent risk
		· Evaluation harness	

Read across a row: the same term may show up in multiple functions when it drives both process and evidence.

FIG A21 · AI RMF crosswalk

pairs with §11 · NIST AI Risk Management Framework mapping.

Sources: NIST AI 100-1

MITRE ATLAS: adversary tactics against AI systems

ATT&CK-style tactics for the AI lifecycle. Each tactic below carries multiple techniques in the live matrix.

RECON	RESOURCE DEV	INITIAL ACCESS	ML MODEL ACCESS
· Search model repos · Search ML datasets	· Acquire ML artifacts · Establish accounts	· ML supply chain · Valid accounts	· Physical env access · Inference API

EXECUTION	PERSISTENCE	DEFENSE EVASION	DISCOVERY
· User execution · Cmd & scripting	· Poison training data · Backdoor ML model	· Evade ML model · Adversarial data	· Discover model family · Discover artifacts

COLLECTION	ML ATTACK STAGE	EXFILTRATION	IMPACT
· ML artifact collect · IS repo data	· Craft adversarial data · Verify attack	· Exfil via inference · Exfil via cyber	· Evade ML model · Denial of service · Cost harvesting · Erode integrity

Use ATLAS to structure red-team scenarios: pick a tactic column, chain techniques from it, verify the impact.

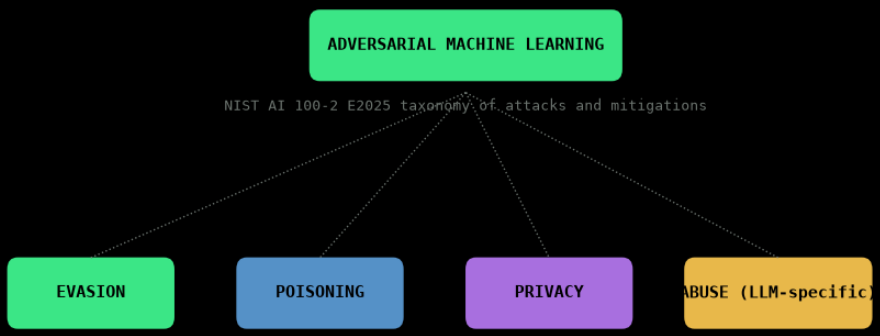
FIG A22 · MITRE ATLAS tactics matrix

pairs with §11 · Threat modeling for AI systems.

Sources: MITRE ATLAS · MITRE ATLAS (GitHub)

NIST AI 100-2 E2025: adversarial ML taxonomy

Four attack classes, each with representative sub-attacks. Shared vocabulary for red teams and auditors.



Pin every 'model attack' discussion to one of these four classes so mitigations map cleanly to a documented mechanism.

FIG A23 · **NIST AI 100-2 taxonomy tree**

pairs with §11 · Shared vocabulary for adversarial ML.

Sources: NIST AI 100-2 E2025

■■■ CHANGE LOG

Version history

v2026.09 (this revision, July 2026). Restored the original 17 figures (A01–A17) from v2026.07 that were dropped during the v2026.08 rebuild. Added six new figures anchored to Section 11: **A18 Adversarial testing four surfaces**, **A19 SHAP vs LIME explanation flow**, **A20 Human-in-the-loop control point**, **A21 AI RMF crosswalk**, **A22 MITRE ATLAS tactics matrix**, and **A23 NIST AI 100-2 adversarial ML taxonomy tree**. Corrected the addendum listing to match the actual v2026.07 figure inventory. Total figure count: 23.

v2026.08 (previous, July 2026). Added **Section 11 — Interpretability, oversight, and adversarial evaluation** covering SHAP family, LIME workflow, surrogate models, human review, and adversarial evaluation. Added Table of Contents. Extended Field Notes to fourteen items. Added 15 sources to the Source Index. Header stamps updated to v2026.08. *Known gap fixed in v2026.09: the visual addendum was left as a text-only listing.*

v2026.07 (original, July 2026). Original ten-section glossary plus Field Notes, Source Index, and Visual Addendum (17 figures). Covered AI/ML foundations, LLM basics, Transformer architecture, local LLM and hardware, retrieval, agents and tool-use, protocols, evaluation, operational framing, and frontier/governance terms.