

Review

Current Trends in Artificial Intelligence Architectures: From Model Scaling to System Intelligence, Post-Transformer Hybrids and World Models

Salvatore Rampone

Department of Law, Economics, Management and Quantitative Methods (DEMM), University of Sannio, 82100 Benevento, Italy; rampone@unisannio.it

Abstract

Artificial intelligence architecture is no longer adequately described by model size alone. Dense Transformers remain the reference architecture for language and multimodal reasoning, but production systems increasingly combine conditional computation, retrieval, memory, tools, verifiers, edge-cloud routing, observability and governance. This review makes three engineering claims. First, sparse Mixture-of-Experts models are currently the clearest capacity-scaling pattern, because they decouple total parameters from active per-token computation, although routing imbalance and distributed communication remain hard constraints. Second, state-space, recurrent and linear attention hybrids are best interpreted as attention-budgeting architectures: they reduce KV-cache and long-context costs, but do not yet displace dense attention in every reasoning regime. Third, JEPA-style latent world models change the learning objective from surface-token or pixel prediction to representation prediction, which is strategically important for perception and planning but still not a drop-in replacement for general language interfaces. To make the maturity claims auditable, this review uses a PRISMA-inspired search protocol, an explicit technology readiness rubric, quantitative comparison tables, hardware and memory-bandwidth analysis, deployment and reproducibility categories, and failure cases for RAG and agents. The main conclusion is that the optimal architecture is task- and constraint-dependent: small dense or hybrid models are often preferred for real-time edge inference, RAG and graph memory for changing enterprise knowledge, frontier dense or sparse models for difficult reasoning, and agentic workflows only when tool permissions, rollback, provenance and human oversight are engineered as first-class components.

Keywords: artificial intelligence; foundation models; post-Transformer architectures; mixture of experts; state-space models; JEPA; world models; retrieval-augmented generation; agentic AI; multimodal AI; liquid neural networks; edge AI; neuro-symbolic AI; AI governance; technology readiness

Academic Editor: Stefano Scanzio

Received: 25 May 2026

Revised: 21 July 2026

Accepted: 23 July 2026

Published: 23 July 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

A useful way to describe the current AI transition is to start from an ordinary production task rather than from a benchmark. Consider an enterprise assistant that must answer a compliance question. A modern version of this system may transcribe speech locally, classify the user request, retrieve policy fragments from a private repository,

check the jurisdiction of the query, call a calculator or workflow API, route difficult reasoning to a larger cloud model, ask a verifier to inspect citations, log the decision path and update memory only if the user has permission. Calling this arrangement a single large language model is no longer accurate. The model remains central, but the architecture is the system of routing, memory, tools, checks and deployment decisions around it.

The central research question is therefore: which parts of contemporary AI intelligence are still located inside the neural model block, and which are being redistributed into memory systems, tool loops, deployment infrastructure and governance layers? The review answers this question by separating mature production patterns from research programs and by evaluating each trend against evidence, efficiency, reproducibility and deployment constraints.

This review therefore treats contemporary AI architecture as a system-level object. The Transformer remains the strongest reference point for general language modeling and multimodal reasoning [1,2]. The scaling-law tradition also remains influential because it showed that larger models, data and compute can improve performance in relatively predictable ways [3,4]. What has changed is the engineering and scientific question. The frontier is less often phrased as “How large can a dense model become?” and more often as “How should computation, memory, perception, action and governance be distributed across a reliable AI system?”

The phrase post-Transformer is used in this article with caution. It does not mean that Transformers are obsolete. It means that a growing fraction of the design space now tries to reduce the exclusive dependence on full attention and token-level autoregression. Some approaches do this in a narrow computational way, for example through state-space models, recurrent hybrids, linear attention, sparse routing or dynamic tokenization. Others do it in a broader cognitive way, most notably through latent world models such as JEPA, where the objective is to predict abstract representations of the world rather than the next surface token [5–9].

The argument of this review is deliberately not symmetrical. MoE, RAG and agentic tool use are already engineering patterns visible in platforms and production APIs [10–16]. JEPA-style world models, neural test-time memory and liquid foundation models are scientifically important but less mature as general-purpose replacements [7–9,17,18]. Treating both groups as equally mature would make the survey neat but misleading.

This review also adds a counterweight: Transformer persistence. Dense and attention-based Transformers continue to dominate because they have stable training recipes, mature serving stacks, broad benchmark coverage, strong in-context learning, large developer ecosystems and extensive alignment and evaluation tooling. The review therefore analyzes post-Transformer work as architectural complementarity rather than as a linear succession narrative.

Figure 1 summarizes the broader system view: user interfaces, agent orchestration, specialized models, memory, reasoning, execution, infrastructure and cross-cutting concerns increasingly form a single architectural stack.

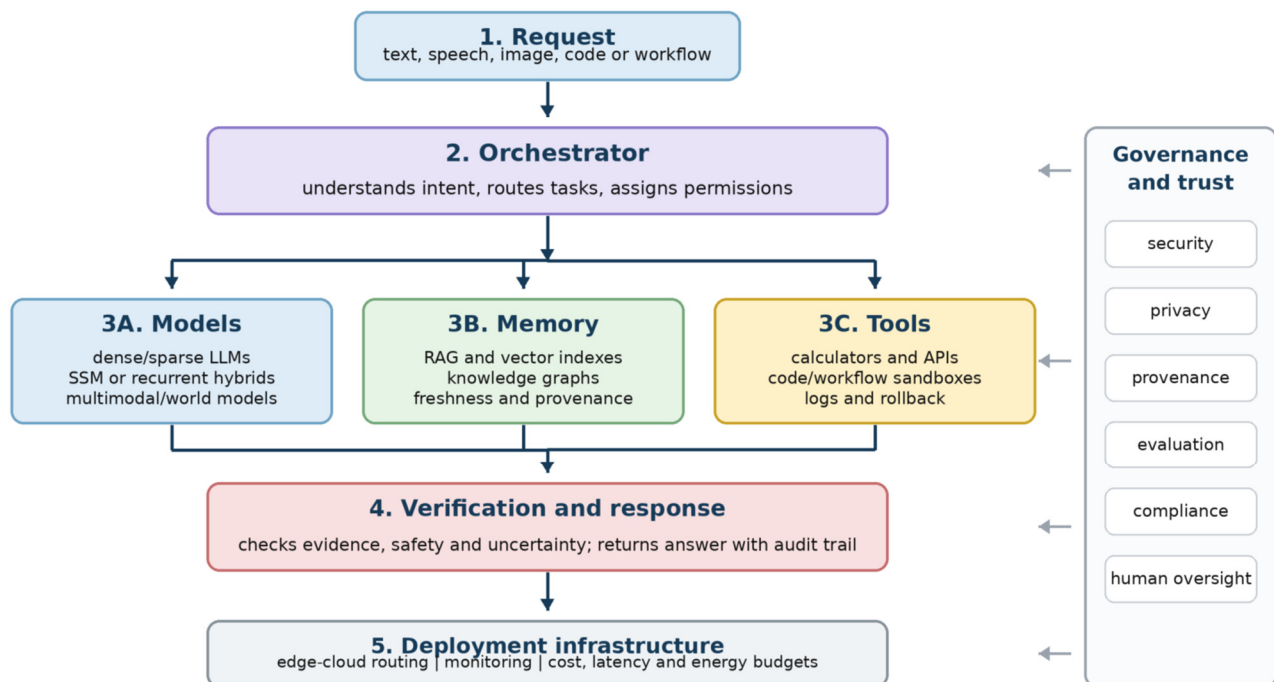


Figure 1. Simplified layered view of contemporary AI system architecture. A user or application request is routed by an orchestrator to model, memory and tool components, then checked by verification, deployment and cross-cutting governance layers before the final response is returned.

2. Materials and Methods

This article is a narrative review and conceptual synthesis of AI architecture trends, adopting a PRISMA-inspired transparency protocol so that the search space, selection logic and evidence categories are auditable. PRISMA is used here as a reporting discipline rather than as a claim that a statistical meta-analysis was performed [19].

The review was updated through June 2026. The final cited corpus contains peer-reviewed papers, arXiv/OpenReview technical reports, model cards, official engineering documentation and policy or standards documents. Searches covered IEEE Xplore, ACM Digital Library, Web of Science Core Collection, Scopus, ACL Anthology, arXiv, OpenReview, major model-provider repositories and official documentation from Meta, Google, OpenAI, Microsoft, Mistral AI, Anthropic, DeepSeek and Liquid AI. Backward and forward snowballing was used from foundational papers such as the Transformer, scaling-law, JEPa, Mamba, Mixtral, Jamba, GraphRAG and ReAct papers.

The selection criteria give more weight to work that changes the architecture of computation or deployment, not merely to work that reports a benchmark improvement. Evidence is separated into five operational indicators: reproducible artifact, independent or cross-family evaluation, production or platform evidence, hardware/inference evidence, and observability/governance evidence. This distinction matters because AI discourse often compresses these categories into a single narrative of breakthrough.

Throughout the article, “architecture” is used broadly. It includes neural blocks such as attention, state-space layers and expert routers; memory mechanisms such as vector stores and neural test-time memory; orchestration patterns such as planner-tool-verifier loops; and deployment choices such as edge-cloud routing and policy enforcement. This broad definition is intentional: the most important AI failures in practice often arise at interfaces between components rather than inside one model block.

Table 1 depicts the PRISMA-inspired search protocol and inclusion criteria used, Table 2 is a technology readiness rubric used to classify architectural trends, and Table 3

is the maturity map. A trend is called production-level only when it has at least a score of 4 on the rubric and at least one official deployment, model card or platform signal.

Table 1. PRISMA-inspired search protocol and inclusion criteria used.

Element	Details
Research questions	RQ1: Which neural architectures reduce exclusive reliance on dense Transformer attention? RQ2: Which system components now carry intelligence outside the base model? RQ3: What evidence maturity and deployment constraints determine whether a trend is production-level or still programmatic?
Core Boolean queries	(“post-Transformer” OR “state space model” OR Mamba OR RetNet OR RWKV OR “linear attention” OR “Mixture of Experts” OR MoE) AND (“foundation model” OR “large language model”); (“retrieval-augmented generation” OR RAG OR GraphRAG OR MemGPT OR “vector database”) AND (production OR enterprise OR evaluation); (“LLM agent” OR “agentic AI” OR ReAct OR Toolformer OR WebArena OR OSWorld OR SWE-bench); (JEPA OR “world model” OR “latent prediction” OR “self-supervised video”) AND (planning OR robot); (“edge AI” OR “on-device” OR “KV cache” OR HBM OR SRAM OR quantization); (“explainable AI” OR “mechanistic interpretability” OR SHAP OR “attention rollout” OR MambaLRP) AND (Transformer OR MoE OR Mamba).
Inclusion criteria	Included records changed architecture, memory, routing, inference, deployment, evaluation or governance; reported quantitative benchmarks or an explicit mechanism; or represented an official production API, model card, engineering report, regulation or standard.
Exclusion criteria	Excluded items were product marketing without technical details, duplicate explainers of the same result, benchmark-only claims without architectural relevance, and application-specific papers not informative for general AI-system architecture.
Screening and synthesis	The final corpus was organized by mechanism, evidence maturity, deployment constraints, reproducibility and risk. Because benchmark protocols differ substantially across sources, quantitative tables report source-reported figures and identify comparability limits rather than claiming a single controlled leaderboard.

Table 2. Technology readiness rubric used to classify architectural trends.

Score	Operational Indicator	Interpretation in This Review
1	Coherent concept but limited reproducible evidence	Strategic research program; useful for research agenda but not production default.
2	Prototype or preprint with narrow benchmarks	Promising mechanism; deployment claims should be cautious.
3	Open implementation or repeated results across several model families	Rapidly maturing; relevant for engineering trials.
4	Official model, API, cloud platform, model card or production-style serving evidence	Production-level direction, with constraints and monitoring requirements.
5	Broad deployment plus mature evaluation, observability and cost/latency evidence	Established system pattern; remaining risks are operational rather than purely experimental.

Table 3. A deliberately asymmetric maturity map.

Trend	Approximate Readiness	Evidence Signal	Main Caution
Dense Transformer scaling	5: established baseline	Llama 3.1 model cards, GPT-style and Gemini-style technical reports, broad benchmark and tooling ecosystem [1,2,13,20,21].	High memory footprint, expensive context, difficult updating and opacity.
Sparse MoE	4: production-level direction	Mixtral, DeepSeek-V3 and Gemini 1.5 expose or publicly discuss MoE-style conditional computation [10–12,14].	Expert imbalance, router stability, all-to-all communication and router interpretability.
SSM/recurrent/linear hybrids	3: rapidly maturing	Mamba, Mamba-2, Jamba and Kimi Linear reduce attention or KV-cache pressure [22–25].	Strong long-context efficiency, but uneven frontier-scale validation and reasoning.
Structured memory and RAG	4: production-level direction	OpenAI File Search, Vertex AI RAG Engine, Microsoft GraphRAG v3.1.1 and Copilot Studio 2026.6.3 show plat-	Poisoning, provenance, stale memory, privacy and retriev-

		form-level patterns [13–16,26,27].	al latency.
Agentic orchestration	4: widely deployed but hard to evaluate	ReAct and Toolformer are research foundations; OpenAI Agents SDK and Anthropic multi-agent systems show engineering patterns [28–31].	Prompt injection, tool misuse, cascading errors and ambiguous success metrics.
JEPAs/world models	2: strategic research program	V-JEPA, V-JEPA 2/2.1 and VL-JEPA show latent prediction for perception and planning [7–9,32].	Not yet a general LLM interface or safety solution.
Edge-native/liquid systems	3: deployment-focused maturation	LFM2 and on-device AI surveys emphasize CPU/NPU latency, memory and packaged deployment [17,33].	Less proven at frontier scale; hardware-specific tuning required.

3. Conceptual and Technical Synthesis

This section synthesizes mechanisms, quantitative evidence, deployment constraints and cross-component relationships.

3.1. End-to-End Case Study: A Governed Compliance Assistant

The running case is an enterprise compliance assistant. A user asks whether a proposed data-transfer workflow is allowed. The system first classifies the request and risk level with a small local model. It then retrieves jurisdiction-specific policy clauses from a vector index and a knowledge graph, checks document version dates, calls a rules engine for threshold calculations, routes hard legal synthesis to a larger model, asks a verifier model to compare the answer against cited sources, and logs the prompt, retrieved documents, tool calls, model versions, uncertainty and human approval status. Memory is updated only if the user has permission and the retention policy allows it.

This case anchors the rest of the synthesis. Dense or sparse foundation models provide language and reasoning; RAG and graph memory provide changing knowledge; tools provide deterministic calculation and workflow execution; agent loops coordinate multi-step work; edge models reduce latency and privacy exposure; and governance makes the resulting autonomy inspectable. A single model can answer some versions of the question, but it cannot by itself guarantee provenance, policy freshness, tool permissioning, rollback or auditability.

3.2. Dense Transformers: Persistence, Not Disappearance

Any discussion of post-Transformer architectures should begin by avoiding a strawman. Full-attention Transformers are not disappearing. Their combination of parallelizable training, scalable representation learning and autoregressive generation remains unusually powerful [1,2]. They also benefit from a mature ecosystem of tokenizers, pre-training recipes, fine-tuning methods, inference optimizations, evaluation suites and developer tools. In many cases, the simplest strong system is still a Transformer-based model connected to retrieval and tools.

Transformer persistence has four practical causes. First, dense models remain the broadest quality baseline for open-ended language and multimodal interaction. Second, serving stacks, quantization libraries, safety filters and evaluation harnesses were built around Transformer inference. Third, long context, grouped-query attention, FlashAttention and KV-cache serving systems have extended the architecture rather than replaced it [13,34,35]. Fourth, developers value predictable behavior, documentation and ecosystem support as much as architectural novelty.

The critique is therefore not that Transformers fail. The critique is that their success has hidden several costs. Attention creates a memory and compute burden for long contexts. Autoregressive next-token prediction is a flexible objective, but it encourages surface-level generation even when the task requires latent state estimation or planning. Very large dense models can be difficult to update, inspect and deploy locally. These

limitations do not invalidate the architecture; they define the boundary conditions under which alternatives become attractive.

3.3. Efficient Scaling and Post-Transformer Sequence Modeling

The first meaning of post-Transformer is computational. It asks whether full attention must be used everywhere, at every layer and for every token. Current systems increasingly answer no. They preserve some of the representational power of the Transformer while reducing the amount of active computation, the size of the KV-cache or the cost of long-context processing.

Mixture-of-Experts (MoE) architectures are the clearest expression of conditional computation. In an MoE layer, a router activates only selected expert subnetworks for each token [36,37]. The model can therefore have large total capacity while using a smaller active subset during inference. The difficulty is not the idea of sparsity; it is the distributed system around it. Routing must remain balanced, experts must not collapse into redundant specialists, and inference must handle communication overhead.

State-space and recurrent hybrids attack a different bottleneck. Mamba introduced selective state spaces, where the state update depends on input content and scales linearly with sequence length [22]. Mamba-2 reframed the relation between state-space models and attention through structured state-space duality [23]. RWKV, RetNet, xLSTM and Griffin pursue related goals: retain parallelizable training while enabling efficient recurrent or recurrence-like inference [38–41]. Jamba makes the hybrid logic explicit by interleaving Transformer and Mamba layers with MoE [24]. The important point is not that one of these blocks has already replaced attention in all settings. It is that attention is now being budgeted rather than assumed.

Linear attention and tokenization research widen the same design question. Kimi Linear uses a hybrid linear attention architecture designed to reduce KV-cache use and improve long-context throughput [25]. Byte Latent Transformer (BLT) challenges the fixed-token assumption by using dynamically sized byte patches, allocating more compute where the next bytes are less predictable and less compute where the sequence is easier [42]. Liquid neural networks and liquid foundation models are best read as deployment-driven alternatives rather than as replacements for all LLMs [17,43].

Table 4 reports a quantitative comparison of representative dense, sparse and hybrid architectures. Values are source-reported and not a single controlled benchmark; they are used to expose trade-offs rather than to declare a leaderboard. Table 5 shows a hardware-aware comparison of major architectural choices. Table 6 evidences scaling and hyperparameter behaviors across architectural families.

Table 4. Quantitative comparison of representative dense, sparse and hybrid architectures.

Model/Family	Scale and Openness	Selected Reported Quantitative Evidence	Efficiency Signal	Best-Fit Interpretation and Caution
Llama 3.1 70B Instruct	Dense optimized Transformer; 70B parameters; 128K context; Llama 3.1 Community License [13].	MMLU 83.6; HumanEval 80.5; 7.0M H100 GPU hours for 70B training [13].	Strong dense baseline with GQA and mature deployment ecosystem.	Best when broad reasoning and tooling support dominate; expensive for local/edge inference.
Mixtral 8x7B	Sparse MoE; 46.7B total, 12.9B active per token; Apache 2.0 [11,12].	Jamba paper comparison reports Mixtral: HellaSwag 86.7, PIQA 83.0, MMLU 70.6, HumanEval 34.8 [24].	Mistral reports 6x faster inference than Llama 2 70B and cost comparable to a 12.9B active model [12].	Good capacity-per-compute example; router and all-to-all communication must be engineered.
Jamba v0.1	Hybrid Transformer-Mamba-MoE; 52B total, 12B active; Apache 2.0 [24].	HellaSwag 87.1, PIQA 83.2, MMLU 67.4, HumanEval 29.3; supports 256K context [24].	At 256K context, Jamba reports 4GB KV cache versus 32GB for Mixtral and up to 3x long-context throughput	Shows attention-budgeting value; quality and stability depend on the attention/Mamba/MoE ratio.

[24].				
DeepSeek-V3	Sparse MoE with MLA; 671B total, 37B active; 128K context; commercial-use weights [10].	Base model: MMLU 87.1, HellaSwag 88.9, PIQA 84.7, HumanEval 65.2; 14.8T tokens and 2.788M H800 GPU hours [10].	Decouples total and active parameters; FP8 and communication co-design reduce training cost [10].	Strong evidence for MoE scaling, but reproduction requires very large infrastructure.
Mamba family	Selective state-space backbone; open research implementation [22].	Mamba abstract reports 5x higher inference throughput than Transformers and linear sequence scaling; Mamba-3B matches Transformers twice its size in reported evaluations [22].	Avoids full attention and recurrently maintains compact state.	Attractive for long/streaming sequences; frontier-scale language reasoning remains less established than Transformers.
LFM2 family	Hybrid gated short convolutions with limited GQA; 350M-8.3B, 32K context; open weights [17].	LFM2-2.6B reports IFEval 79.56 and GSM8K 82.41; LFM2 family reports up to 2x CPU prefill/decode versus similar-size models [17].	Designed through hardware-in-the-loop search for CPU/NPU/edge constraints.	Best interpreted as deployment-first architecture; not a replacement for frontier reasoning models.

Table 5. Hardware-aware comparison of major architectural choices.

Architecture Choice	Dominant Bottleneck	Training Implication	Inference Implication	Practical Mitigation
Dense full attention	Quadratic attention matrix and HBM traffic for attention activations; linear KV cache in decoding.	Training speed depends on IO-aware kernels such as FlashAttention [34].	Decode latency and memory become dominated by weights plus KV cache [35,44].	GQA/MQA, FlashAttention, PagedAttention, quantization and batching.
Sparse MoE	Router decisions, expert imbalance and all-to-all communication.	More total capacity at similar active FLOPs, but distributed training is communication-sensitive.	Only top experts run per token, but serving must place experts and route batches efficiently.	Auxiliary or auxiliary-free load balancing, expert parallelism, router monitoring.
SSM/recurrent hybrid	State update kernel efficiency and information bottleneck in compact state.	Parallel scan and custom kernels needed for throughput.	Small recurrent state can reduce long-context memory.	Hybridize with attention layers; benchmark state tracking and low-salience retrieval.
Linear attention/KV reduction	Approximation quality, kernel design and cache management.	May reduce asymptotic cost but not always wall-clock time.	Can improve long-context throughput when cache bandwidth is limiting.	Use IO-aware kernels and evaluate with realistic long documents.
RAG/GraphRAG	Embedding, vector/graph retrieval, reranking and context insertion.	Index building adds batch cost outside model training.	Retrieval adds latency but avoids retraining for knowledge updates.	Caching, metadata filters, hybrid search, reranking and provenance checks.
Edge/low-code/serverless deployment	Cold starts, memory ceiling, accelerator availability and network round trips.	Architecture search must optimize model size and target hardware.	Small local models may beat large cloud models on latency/privacy.	Distillation, quantization, on-device routing, serverless warm pools.

Table 6. Scaling and hyperparameter behaviors across architectural families.

Family	Main Scaling Variables	Expected Benefit	Failure Mode to Evaluate
Dense Transformer	Parameter count, tokens, context length, depth/width, attention heads, GQA/MQA choices.	Predictable quality gains under compute/data scaling.	Expensive context, memorization, stale knowledge, opacity.
MoE Transformer	Total experts, active experts top-k, router temperature, expert size, load-balance losses, expert parallelism.	More capacity at fixed active compute.	Expert collapse, route instability, communication bottlenecks.
SSM/recurrent hybrid	State size, scan kernel, attention-to-SSM ratio, convolution width, normalization.	Long-context or streaming efficiency.	Loss of rare details, weaker in-context learning if attention is removed too aggressively.
RAG/memory	Chunk size, overlap, embedding dimension, top-k, reranker,	Updatable knowledge and lower hallucination when sources are	Wrong source, stale source, poisoning, privacy leakage.

	metadata filters, retention policy.	correct.	
Agentic workflow	Planning horizon, tool schema, permission model, verifier threshold, stop condition.	Multi-step execution and external action.	Cascading tool errors, prompt injection, hidden unsafe intermediate actions.
World model/JEPA	Latent target definition, masking ratio, encoder/predictor capacity, action-conditioning data.	Prediction of abstract future state and planning support.	Poor language interface, weak causal validation, uncertain safety.

Figure 2 depicts sparse MoE routing in one layer. A router selects a small top-k set of experts for each token; only those experts run, and their weighted outputs form the next representation.

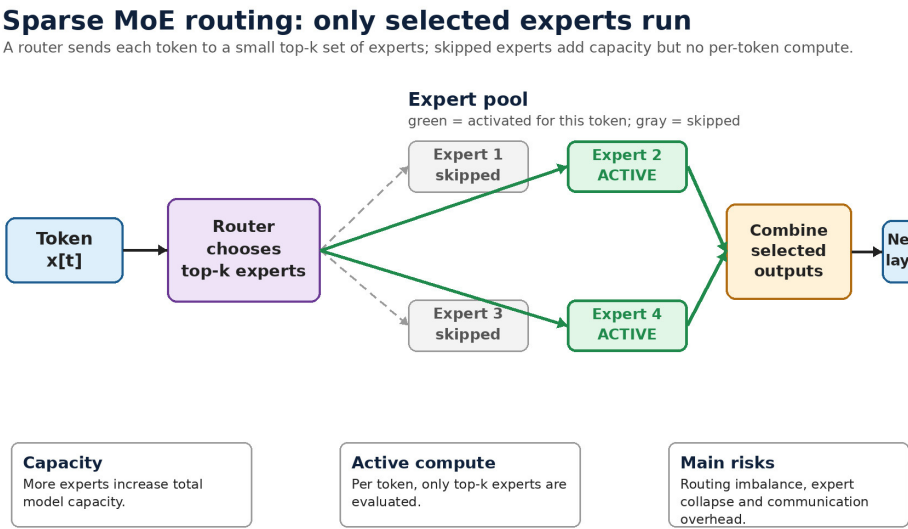


Figure 2. Sparse MoE routing in one layer.

Figure 3 shows selective SSM/recurrent update. Each token updates a compact hidden state, which is carried to the next time step; inference stores this state rather than a full-attention KV cache.

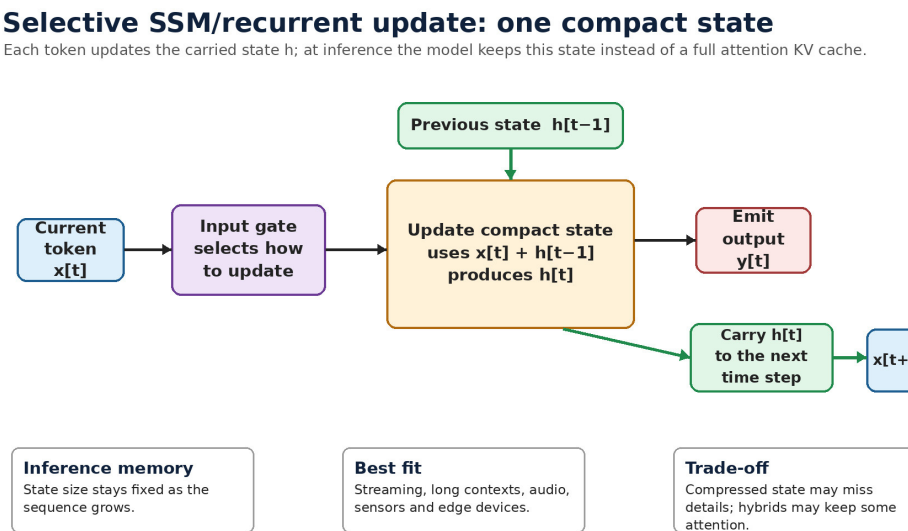


Figure 3. Selective SSM/recurrent update.

3.4. Memory and Retrieval: Not a Longer Prompt, but a Managed Resource

Retrieval-augmented generation (RAG) separated knowledge access from language generation by retrieving external text and conditioning generation on it [45]. That basic pattern remains useful, but it is too weak for many real systems. A legal assistant, for example, may need semantic memory for definitions, episodic memory for previous user interactions, procedural memory for filing steps, graph memory for entities and jurisdictions, and a retention policy that prevents personal data from persisting indefinitely. In such a system, “context” is not a string appended to a prompt. It is an engineered resource with ownership, provenance, expiry and permissions.

GraphRAG makes this point concrete. Instead of retrieving only local passages, a system can build an entity graph, cluster related entities and generate community summaries to answer global questions about a corpus [26]. MemGPT describes context management as an operating-system problem, where a model manages memory tiers under a limited context window [46]. Titans extends the theme by treating attention as short-term memory and adding neural long-term memory that learns what to store at test time [18].

These examples show why a larger context window alone is not a memory architecture. The system needs metadata filters, source dates, provenance display, retrieval evaluation and a verifier that checks whether the cited source actually supports the answer. In the running compliance case, a final answer is considered valid only if the cited policy version, jurisdiction and calculation trace are logged with the response.

Table 7 resumes RAG and memory overheads.

Table 7. RAG and memory overheads that should be reported in engineering reviews.

Stage	Cost or Latency Driver	Failure Example	Engineering Control
Ingestion and embedding	Document parsing, chunking, embedding model size, batch throughput and index writes.	A policy PDF is chunked across an exception clause, so retrieval returns the rule but not the exception.	Chunking tests, semantic boundary detection, document-version metadata.
Retrieval	Vector search, hybrid keyword search, graph traversal, top-k and reranking.	A legally correct paragraph from the wrong jurisdiction outranks the current local policy.	Jurisdiction/date filters, freshness scores, source whitelists.
Context insertion	More retrieved text increases prompt tokens and attention/KV cost.	The right chunk is retrieved but buried under low-salience text and ignored.	Reranking, source compression, citation-aware prompting, verifier checks.
Persistent user memory	Personalization improves response quality but introduces retention and consent risk.	User preference memory stores sensitive data longer than policy allows.	Consent flags, edit/delete UI, TTL, access control and audit logs.
Adversarial retrieval	Documents can contain hidden instructions or malicious content.	A retrieved page instructs the agent to ignore policy and exfiltrate data.	Prompt-injection filters, tool separation, untrusted-content boundaries.

3.5. Agents: Prompts Become Controlled Workflows

Agentic AI is the system-level expression of tool use, memory and planning. An agent is not merely a chat interface. It is a loop that can propose a plan, call tools, observe results, revise its strategy, maintain state and decide when to stop. ReAct showed that interleaving reasoning and action can improve performance in interactive environments [28]. Toolformer showed that language models can learn when and how to call APIs [29]. These ideas now appear in coding agents, data-analysis assistants, research agents and workflow automation systems.

A concrete coding agent illustrates the architectural difference. It does not simply generate a patch. It reads the repository, identifies relevant files, writes a change, runs tests in a sandbox, observes failures, edits again, produces a diff and asks for human approval before merging. The LLM provides planning and language/code generation,

but the reliability comes from the surrounding loop: file permissions, test execution, logs, rollbacks and review gates.

The most serious agent problems are coordination problems. Multi-agent systems can create conflicting goals, redundant work, hidden state drift and cascading errors. Tool use introduces security risks, especially prompt injection and over-broad permissions. In the compliance case, an agent that can read policies should not automatically have permission to email external recipients, change workflow settings or update long-term memory. Mature agent architecture therefore resembles a governed workflow engine with neural planners, not an autonomous black box.

Table 8 shows agent evaluation examples.

Table 8. Agent evaluation examples added to make the risks quantitative.

Benchmark or Evidence Source	Task Type	Reported or Recommended Metric	Interpretation for Architecture
WebArena [47]	Long-horizon web tasks over realistic sites.	Best GPT-4-based agent reported 14.41% task success versus 78.24% human success.	Agent success is still far from human reliability on realistic web tasks; evaluation must include process traces.
OSWorld [48]	369 real computer-use tasks across desktop/web applications.	Execution-based task success; screen/action trajectories.	Evaluates GUI perception, action selection and multi-application workflows.
AgentBench [49]	Eight interactive environments for LLM-as-agent evaluation.	Success/reward by environment; failure modes include long-term reasoning and instruction following.	Useful for comparing agent scaffolds, not only base models.
SWE-bench Verified [50]	500 human-validated software issues.	Percent resolved under a fixed harness.	Coding agents should report patch validity, test pass rate, attempts, tool calls and roll-back quality.
Production workflow audit	Enterprise agents with tools/APIs.	Tool-call error rate, unsafe action rate, average steps, human override rate and incident recovery time.	A good final answer can hide unsafe intermediate actions unless logs are evaluated.

3.6. Multimodality and World Models: The Stronger Post-Transformer Hypothesis

The second and more ambitious meaning of post-Transformer is not about the cost of attention. It is about the learning objective. CLIP showed the value of aligning image and text representations [51]. GPT-4, Gemini and Chameleon extended the foundation-model paradigm toward multimodal inputs and mixed-modal generation [20,21,52]. But these systems still often operate through tokenized interfaces. World-model research asks a different question: can a system learn latent structure, dynamics and possible futures from perception?

JEPA-style learning differs from next-token prediction in the prediction target. Autoregressive language modeling estimates the distribution of the next surface symbol conditioned on previous symbols. A JEPA model predicts the representation of a target region from a context representation, deliberately avoiding reconstruction of unpredictable surface details [5,6]. This can make the objective more aligned with semantic state estimation, but it also creates limitations: the learned latent space must be probed, decoded or coupled to a planner before it can serve as a general user-facing interface.

V-JEPA demonstrated latent feature prediction for video without labels, text supervision, negative examples or pixel reconstruction [7]. V-JEPA 2 scaled the approach to internet-scale video and added a small amount of robot interaction data, with the aim of supporting understanding, prediction and planning in the physical world [8]. V-JEPA 2.1 moved toward denser video features and stronger spatial-temporal grounding [9]. VL-JEPA extended the idea to vision-language by predicting continuous text embeddings rather than generating text autoregressively at every step [32].

There is a boundary condition: self-supervised latent prediction is not always superior to supervised learning. Supervised architectures are advantageous when labels are reliable, targets are narrow, and auditability matters, for example in industrial segmentation, medical triage rules or damage detection. Their weakness is dependence on labeled data and reduced robustness under domain shift. Self-supervised approaches scale to unlabeled perceptual streams and may learn broader representations, but their objectives can be harder to interpret, validate and align with task-specific constraints.

This direction should be interpreted cautiously. JEPA-style models are not yet general replacements for language models, nor do they by themselves solve planning, embodiment or safety. Their importance is conceptual: they make the post-Transformer question less about replacing attention and more about replacing next-token prediction as the dominant explanatory frame for intelligence.

This conceptual shift is depicted in Figure 4.

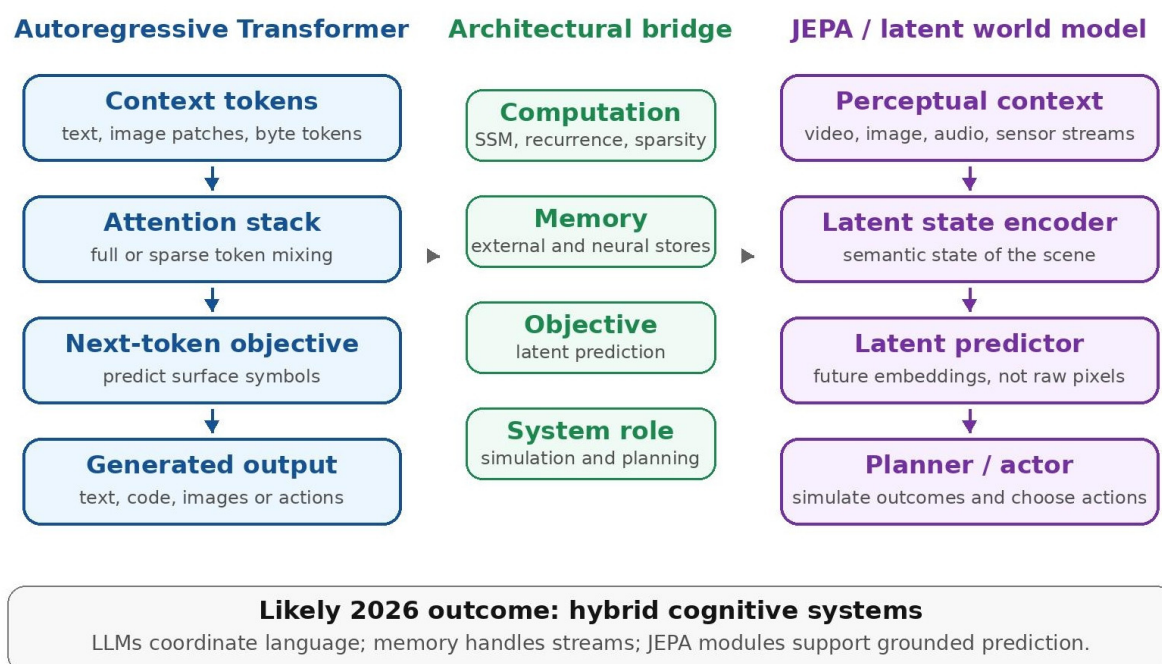


Figure 4. Conceptual shift from surface-level autoregression to latent world modeling. In practice, near-term systems are likely to combine these columns rather than choose only one.

3.7. Reasoning, Meta-Cognition and Neuro-Symbolic Constraints

Reasoning remains one of the least stable terms in contemporary AI. Some systems improve because they are trained to produce better intermediate chains; others improve because they spend more computation at inference time, search over candidates, use verifiers or distill reasoning traces into smaller models [53,54]. For architecture, the important point is that output generation is becoming less direct. The system increasingly follows a generate-check-act pattern.

Meta-cognitive mechanisms make this pattern explicit. A system can estimate uncertainty, decide when to retrieve, compare candidate answers, ask for human confirmation, run a verifier, refuse unsafe tool use or fall back to a more capable model. These behaviors should not be treated as optional user-interface features. In high-stakes contexts, they are part of the architecture of reliability.

Neuro-symbolic integration addresses a related limitation. Neural models are flexible but weak at explicit constraints, provenance and formal guarantees. Symbolic systems, knowledge graphs, planners and constraint solvers are less flexible but more in-

spectable. Recent work on neuro-symbolic AI and LLM-knowledge graph integration suggests that hybrid architectures will be important in domains where rules, entities and auditability matter [55,56]. A medical triage assistant, for instance, should not rely only on generated text. It should also check structured contraindications, guideline dates, patient-specific restrictions and escalation rules.

3.8. Transparency and Explainability Across New Architectures

As architectures move from single dense models to MoE routers, SSM states, RAG memories and multi-agent tools, transparency also becomes system-level. Attention maps alone cannot explain why a router selected an expert, why a state-space model retained a hidden state, why a retriever returned a stale document or why an agent called a tool. Explainability must therefore combine model-internal methods with provenance and process logs. Table 9 outlines XAI and observability mechanisms for dense, sparse, recurrent and system-level architectures

Table 9. XAI and observability mechanisms for dense, sparse, recurrent and system-level architectures.

Architecture	Candidate Explanation Method	What it Can Reveal	Limitation
Dense Transformer	Attention rollout, gradient/activation attribution, SHAP-style feature attribution, sparse autoencoders and mechanistic circuits [57–59].	Token, head, feature or circuit contributions to an output.	Attention is not explanation by itself; high-dimensional features remain hard to validate.
MoE	Router attribution, expert activation histograms, expert specialization probes, counterfactual expert ablation.	Which experts were used and whether routing is balanced or interpretable.	Experts may not correspond to human semantic categories; routing may be input-distribution sensitive.
SSM/Mamba	Layer-wise relevance propagation and state-influence analysis such as MambaLRP [60].	Which inputs affect recurrent state and output through time.	Compact state can hide low-salience information; methods are less mature than Transformer XAI.
RAG/memory	Source provenance, retrieval traces, freshness metadata, citation-verification scores.	Whether the answer is grounded in the retrieved source.	A cited source may be relevant but insufficient or outdated.
Agents/tools	Tool-call logs, permission traces, sandbox outputs, rollback records, human approvals.	Whether the process was safe, reversible and authorized.	Logs can be incomplete or too verbose without process metrics.
World models	Latent-space probes, counterfactual rollouts, action-conditioned prediction tests.	Whether latent states support prediction and planning.	Good latent prediction does not guarantee causal understanding or safe action.

3.9. Edge-Cloud, Federated and Energy-Aware AI

Deployment increasingly shapes architecture. Not all inference can move to a centralized cloud. On-device and edge models reduce latency, protect privacy, support offline operation and control cost. Cloud models remain necessary for heavy reasoning, large-scale retrieval, training and access to specialized accelerators. The result is a placement problem: decide which part of the task runs on the device, which at the edge and which in the cloud.

This placement problem changes the value of architectural innovations. Quantization, pruning, distillation, sparsity, KV-cache reduction, short-convolution blocks and hardware-aware search are not afterthoughts. They determine whether a model can run in a vehicle, on a phone, in a factory robot or inside a regulated enterprise network. Surveys of on-device AI and federated LLMs emphasize latency, privacy, communication limits, heterogeneous data and hardware acceleration as central constraints [33,61].

Low-code and serverless platforms add a different constraint. They make RAG and agent deployment easier, but cold starts, rate limits, hidden orchestration logic and vendor-specific tool interfaces can obscure reproducibility. For the running compliance assistant, a serverless retrieval function may be economically attractive, but audit requirements may force private indexing, deterministic reranking and explicit logging of every tool call. The right architecture is therefore not always the largest model; it is the placement that satisfies latency, privacy, reversibility and compliance, as evidenced in Table 10.

Table 10. Deployment-oriented decision matrix.

Use Case Constraint	Preferred Pattern	Why	Caution
Real-time local interaction	Small dense, distilled, quantized or edge-first hybrid model.	Lowest latency, offline operation and privacy.	May need cloud fallback for difficult reasoning.
Changing enterprise knowledge	RAG, GraphRAG and governed memory.	Updates knowledge without re-training the base model.	Requires freshness, access control and citation verification.
Very long documents	Hybrid attention-SSM, KV-cache optimization or retrieval-first decomposition.	Avoids simply paying full-attention cost over all tokens.	Need realistic synthesis benchmarks, not only needle retrieval.
High-consequence workflow	Agent with typed tools, sandbox, verifier, rollback and human approval.	Enables action while constraining unsafe autonomy.	More components increase failure surface.
Physical-world planning	JEPA/world model plus planner and safety layer.	Latent prediction may support grounded simulation.	Still early; not a substitute for safety validation.

3.10. Governance, Evaluation and Observability as Architecture

As AI systems become more agentic and interconnected, governance is no longer outside the architecture. Security, privacy, safety, observability, evaluation and compliance must be present in the design of memory, tools, routing and execution. An agentic system should log retrieved documents, tool calls, model versions, prompts, permissions, intermediate decisions and human interventions. A memory system should implement retention, deletion and access-control policies. A model portfolio should define routing, fallback and regression monitoring.

Standards and regulations reinforce this view. The NIST AI Risk Management Framework organizes trustworthy AI around govern, map, measure and manage functions [62]. The European Union AI Act establishes a risk-based legal framework for AI systems in the EU [63]. Regardless of jurisdiction, such frameworks push technical systems toward traceability, testing and accountability. In practice, this means moving from benchmark-only evaluation to continuous monitoring: task success, hallucination rate, retrieval grounding, tool safety, privacy leakage, cost, latency and incident recovery (see Tables 11 and 12).

Table 11. Architecture-aware evaluation criteria.

Design Decision	A Superficial Metric	A More Architectural Metric	Why It Matters
Use a larger model	Benchmark accuracy	Accuracy per dollar, joule and latency under realistic context	A larger model may be unusable at the required placement
Add RAG	Retrieved chunk relevance	Answer faithfulness, source provenance and freshness	A relevant chunk can still support a wrong answer
Deploy an agent	Task completion rate	Tool-call safety, rollback behavior and human override quality	Unsafe intermediate actions can be hidden by a good final answer
Use long context	Needle-in-haystack score	Synthesis, contradiction resolution and attention to low-salience evidence	Synthetic retrieval does not equal real analysis
Store user memory	Personalization accuracy	Consent, retention, editability and leakage resistance	Memory improves usability but creates governance risk

Use MoE routing	Total parameter count.	Active parameters, routing entropy, expert load and cross-node communication cost.	Sparse capacity can become inefficient or opaque.
-----------------	------------------------	--	---

Table 12. Reproducibility and openness categories for representative architectures.

Category	Examples	Reproducibility Status	Compute Barrier and Caveat
Open or available weights with technical report	Mixtral, Jamba, Llama 3.1, DeepSeek-V3, LFM2 [10,11,13,17,24].	Weights or model cards support independent inference and some evaluation.	Full pretraining is still often beyond academic compute budgets.
Open method with code but expensive data/compute	GraphRAG, Mamba, JEPA variants [6,7,22,26].	Mechanisms can be tested; scale may be hard to match.	Benchmark claims should separate small-scale replication from frontier-scale performance.
Closed API or limited technical disclosure	Some frontier LLMs and video/world-model systems [20,21].	Useful as production evidence but weak for architecture-level reproduction.	Review claims should distinguish product behavior from verified architecture.
Standards and governance frameworks	NIST AI RMF and EU AI Act [62,63].	Reproducible as evaluation/control requirements, not model artifacts.	Implementation details depend on domain and jurisdiction.

4. Discussion—Open Questions and Boundary Conditions

The AI field is diversifying rather than converging on a single replacement architecture. This creates several research questions.

- Fair comparison requires dense, sparse, recurrent, state-space, liquid and hybrid models to be compared under matched compute, memory, latency, data and hardware constraints.
- Long-context evaluation should move beyond synthetic needle tasks toward synthesis, contradiction resolution, multi-document reasoning and provenance checking.
- Memory governance requires mechanisms for forgetting, consent, privacy, poisoning resistance, source attribution and user control.
- Safe autonomy requires typed tools, permission models, sandboxing, interpretable logs, stop conditions and recovery protocols.
- World-model validation should test whether latent representations support causal prediction and planning, not only downstream classification.

There are also boundary conditions for the thesis of this article. Some applications still benefit most from a single strong model and a simple interface. Some post-Transformer proposals may remain niche. Some memory and agent systems may add more failure modes than they remove. The system-centric view should therefore be treated as a design hypothesis, not as an excuse to make every application unnecessarily complex.

The revised limitations are practical. MoE can be inefficient when batches are small or expert placement is poor; a workaround is router monitoring and expert-aware serving. SSMs can compress away rare facts; a workaround is hybrid attention layers and retrieval. RAG can retrieve stale or adversarial content; a workaround is metadata filtering, document provenance and citation verification. Agents can cascade errors; a workaround is typed tools, small permissions, sandboxing, human approval and rollback. World models can learn useful latent dynamics without providing a safe action policy; a workaround is separate planning validation and conservative control.

Nevertheless, the direction is hard to ignore. The decisive innovations are increasingly found at interfaces: between attention and recurrence, model and memory, language and perception, planner and tool, cloud and device, autonomy and governance. A research agenda focused only on the neural block will miss many of the systems that users actually experience.

5. Conclusions

The current direction of AI architecture is best described as a transition from model scale to system intelligence. Dense Transformers remain foundational and persistent, but the most useful engineering question is now constraint-sensitive placement: which component should solve which part of the task under latency, memory, cost, energy, privacy, provenance and safety requirements?

The hard technical takeaways are as follows. For real-time edge tasks, smaller dense or edge-first hybrid models often dominate because locality, latency and privacy matter more than absolute benchmark performance. For long context, SSM/recurrent hybrids, linear attention and KV-cache management are strongest when the task rewards streaming or memory efficiency, but dense attention remains valuable for global mixing and complex in-context reasoning. For capacity scaling under compute limits, sparse MoE is the clearest production-level pattern, provided routing, expert load and communication are monitored. For changing knowledge, RAG and GraphRAG are preferable to retraining, but only with freshness, provenance and poisoning controls. For complex workflows, agentic systems are useful only when tools are typed, permissions are narrow, logs are inspectable and human override is available. For grounded perception and planning, JEPA-style world models are among the most important strategic research programs, but they are not yet general language-model replacements.

Future research should therefore prioritize apples-to-apples architecture benchmarks under matched hardware constraints, quantitative reporting of active FLOPs and memory bandwidth, XAI methods for routers and recurrent states, realistic long-horizon agent evaluation, open-weight reproducibility for world models, and governance metrics that measure process quality rather than final-answer accuracy alone.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study.

Acknowledgments: During the preparation of this manuscript, the author used Grok 4 to gather some details and ChatGPT GPT-5.5 Pro for preliminary structural organization. The author reviewed and edited the results and takes full responsibility for the content of this publication.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
API	Application Programming Interface
BLT	Byte Latent Transformer
CLIP	Contrastive Language-Image Pre-training
GQA	Grouped-Query Attention
HBM	High-Bandwidth Memory
JEPA	Joint Embedding Predictive Architecture
KV	Key-Value
LFM	Liquid Foundation Model
LLM	Large Language Model
LTC	Liquid Time-Constant Network
MoE	Mixture-of-Experts
RAG	Retrieval-Augmented Generation
SRAM	Static Random-Access Memory
SSM	State Space Model
TRL	Technology Readiness Level
XAI	Explainable Artificial Intelligence

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30*; NeurIPS: San Diego, CA, USA, 2017.
2. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33*; NeurIPS: San Diego, CA, USA, 2020.
3. Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T.B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; Amodei, D. Scaling laws for neural language models. *arXiv* **2020**, arXiv:2001.08361.
4. Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; Casas, D.D.L.; Hendricks, L.A.; Welbl, J.; Clark, A.; et al. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems 35*; NeurIPS: San Diego, CA, USA, 2022.
5. LeCun, Y. *A Path Towards Autonomous Machine Intelligence*; version 0.9.2; OpenReview: Amherst, United States, 2022.
6. Assran, M.; Duval, Q.; Misra, I.; Bojanowski, P.; Vincent, P.; Rabbat, M.; LeCun, Y.; Ballas, N. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, 17–24 June 2023; pp. 15619–15629.
7. Bardes, A.; Garrido, Q.; Ponce, J.; Chen, X.; Rabbat, M.; LeCun, Y.; Assran, M.; Ballas, N. Revisiting feature prediction for learning visual representations from video. *arXiv* **2024**, arXiv:2404.08471.
8. Assran, M.; Bardes, A.; Fan, D.; Garrido, Q.; Howes, R.; Muckley, M.; Rizvi, A.; Roberts, C.; Sinha, K.; Zhoulus, A.; et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv* **2025**, arXiv:2506.09985.
9. Mur-Labadia, L.; Muckley, M.; Bar, A.; Assran, M.; Sinha, K.; Rabbat, M.; LeCun, Y.; Ballas, N.; Bardes, A. V-JEPA 2.1: Unlocking dense features in video self-supervised learning. *arXiv* **2026**, arXiv:2603.14482.
10. DeepSeek-AI. DeepSeek-V3 technical report. *arXiv* **2024**, arXiv:2412.19437.
11. Jiang, A.Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D.S.; Casas, D.D.L.; Hanna, E.B.; Bressand, F.; et al. Mixtral of experts. *arXiv* **2024**, arXiv:2401.04088.
12. Mistral AI. Mixtral of Experts. Official Research Blog. 2023. Available online: <https://mistral.ai/news/mixtral-of-experts/> (accessed on 20 June 2026).
13. Meta. Llama 3.1 Model Card. GitHub Repository. 2024. Available online: <https://github.com/meta-llama/llama-models/> (accessed on 20 June 2026).
14. Google. Introducing Gemini 1.5, Google’s Next-Generation AI Model. Official Blog. 2024. Available online: <https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/> (accessed on 20 June 2026).
15. OpenAI. File Search. OpenAI API Documentation. 2026. Available online: <https://developers.openai.com/> (accessed on 20 June 2026).
16. Google Cloud. Vertex AI RAG Engine Overview. Google Cloud Documentation. 2026. Available online: <https://cloud.google.com/vertex-ai/> (accessed on 20 June 2026).
17. Amini, A.; Banaszak, A.; Benoit, H.; Böök, A.; Dakhra, T.; Duong, S.; Eng, A.; Fernandes, F.; Härkönen, M.; Harrington, A.; et al. LFM2 technical report. *arXiv* **2025**, arXiv:2511.23404.
18. Behrouz, A.; Zhong, P.; Mirrokni, V. Titans: Learning to memorize at test time. *arXiv* **2025**, arXiv:2501.00663.
19. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ* **2021**, *372*, n71.
20. OpenAI. GPT-4 technical report. *arXiv* **2023**, arXiv:2303.08774.
21. Gemini Team. Gemini: A family of highly capable multimodal models. *arXiv* **2023**, arXiv:2312.11805.
22. Gu, A.; Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv* **2023**, arXiv:2312.00752.
23. Dao, T.; Gu, A. Transformers are SSMS: Generalized models and efficient algorithms through structured state space duality. *arXiv* **2024**, arXiv:2405.21060.
24. Lenz, B.; Lieber, O.; Araz, A.; Bergman, A.; Manevich, A.; Peleg, B.; Aviram, B.; Almagor, C.; Fridman, C.; Padnos, D.; et al. Jamba: Hybrid Transformer-Mamba language models. In *Proceedings of the International Conference on Learning Representations*, Singapore, 24–28 April 2025.
25. Team, K.; Zhang, Y.; Lin, Z.; Yao, X.; Hu, J.; Meng, F.; Liu, C.; Men, X.; Yang, S.; Li, Z.; et al. Kimi Linear: An expressive, efficient attention architecture. *arXiv* **2025**, arXiv:2510.26692.
26. Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; Larson, J. From local to global: A Graph RAG approach to query-focused summarization. *arXiv* **2024**, arXiv:2404.16130.

27. Microsoft Research. GraphRAG: New Tool for Complex Data Discovery Now on GitHub. Microsoft Research Blog. 2024. Available online: <https://www.microsoft.com/en-us/research/> (accessed on 20 June 2026).
28. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing reasoning and acting in language models. *arXiv* **2023**, arXiv:2210.03629.
29. Schick, T.; Dwivedi-Yu, J.; Dessi, R.; Raileanu, R.; Lomeli, M.; Zettlemoyer, L.; Cancedda, N.; Scialom, T. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36*; NeurIPS: San Diego, CA, USA, 2023.
30. OpenAI. Agents SDK Tools Documentation. OpenAI GitHub Pages. 2026. Available online: <https://openai.github.io/openai-agents-python/> (accessed on 20 June 2026).
31. Anthropic. How We Built Our Multi-Agent Research System. Engineering Blog. 2025. Available online: <https://www.anthropic.com/engineering/> (accessed on 20 June 2026).
32. Chen, D.; Shukor, M.; Moutakanni, T.; Chung, W.; Yu, L.; Kasarla, T.; Bolourchi, A.; LeCun, Y.; Fung, P. VL-JEPA: Joint embedding predictive architecture for vision-language. *arXiv* **2025**, arXiv:2512.10942.
33. Wang, X.; Tang, Z.; Guo, J.; Meng, T.; Wang, C.; Wang, T.; Jia, W. Empowering edge intelligence: A comprehensive survey on on-device AI models. *arXiv* **2025**, arXiv:2503.06027.
34. Dao, T.; Fu, D.Y.; Ermon, S.; Rudra, A.; Re, C. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*; 2022.
35. Kwon, W.; Li, Z.; Zhuang, S.; Sheng, Y.; Zheng, L.; Yu, C.H.; Gonzalez, J.E.; Zhang, H.; Stoica, I. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, Koblenz, Germany, 23–26 October 2023.
36. Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv* **2017**, arXiv:1701.06538.
37. Fedus, W.; Zoph, B.; Shazeer, N. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.* **2022**, *23*, 1–39.
38. Peng, B.; Alcaide, E.; Anthony, Q.; Albalak, A.; Arcadinho, S.; Biderman, S.; Cao, H.; Cheng, X.; Chung, M.; Derczynski, L.; et al. RWKV: Reinventing RNNs for the Transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*; Association for Computational Linguistics: Singapore, 2023.
39. Sun, Y.; Dong, L.; Huang, S.; Ma, S.; Xia, Y.; Xue, J.; Wang, J.; Wei, F. Retentive Network: A successor to Transformer for large language models. *arXiv* **2023**, arXiv:2307.08621.
40. Beck, M.; Poppel, K.; Spanring, M.; Auer, A.; Prudnikova, O.; Kopp, M.; Klambauer, G.; Brandstetter, J.; Hochreiter, S. xLSTM: Extended long short-term memory. In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2024.
41. De, S.; Smith, S.L.; Fernando, A.; Botev, A.; Cristian-Muraru, G.; Gu, A.; Haroun, R.; Berrada, L.; Chen, Y.; Srinivasan, S.; et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv* **2024**, arXiv:2402.19427.
42. Pagnoni, A.; Pasunuru, R.; Rodriguez, P.; Nguyen, J.; Muller, B.; Li, M.; Zhou, C.; Yu, L.; Weston, J.E.; Zettlemoyer, L.; et al. Byte Latent Transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*; Association for Computational Linguistics: Vienna, Austria, 2025; Volume 1: Long Papers, pp. 9238–9258. <https://doi.org/10.18653/v1/2025.acl-long.453>.
43. Hasani, R.; Lechner, M.; Amini, A.; Rus, D.; Grosu, R. Liquid time-constant networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*; Association for the Advancement of Artificial Intelligence: Washington, DC, USA, 2021; Volume 35, pp. 7657–7666. <https://doi.org/10.1609/aaai.v35i9.16936>.
44. Pope, R.; Douglas, S.; Chowdhery, A.; Devlin, J.; Bradbury, J.; Heek, J.; Xiao, K.; Agrawal, S.; Dean, J. Efficiently scaling Transformer inference. In *Proceedings of the Machine Learning and Systems*, Miami, FL, USA, 4–8 June 2023.
45. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.T.; Rocktäschel, T.; et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33*; NeurIPS: San Diego, CA, USA, 2020.
46. Packer, C.; Wooders, S.; Lin, K.; Fang, V.; Patil, S.G.; Stoica, I.; Gonzalez, J.E. MemGPT: Towards LLMs as operating systems. *arXiv* **2023**, arXiv:2310.08560.
47. Zhou, S.; Xu, F.F.; Zhu, H.; Zhou, X.; Lo, R.; Sridhar, A.; Cheng, X.; Ou, T.; Bisk, Y.; Fried, D.; et al. WebArena: A realistic web environment for building autonomous agents. In *Proceedings of the International Conference on Learning Representations*, Vienna, Austria, 7–11 May 2024.

48. Xie, T.; Zhang, D.; Chen, J.; Li, X.; Zhao, S.; Cao, R.; Hua, T.; Cheng, Z.; Shin, D.; Lei, F.; et al. OSWorld: Benchmarking multi-modal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems 37*; NeurIPS: San Diego, CA, USA, 2024.
49. Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; et al. AgentBench: Evaluating LLMs as agents. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024.
50. Jimenez, C.E.; Yang, J.; Wettig, A.; Yao, S.; Pei, K.; Press, O.; Narasimhan, K. SWE-bench: Can language models resolve real-world GitHub issues? In Proceedings of the International Conference on Learning Representations, Vienna Austria, 7–11 May 2024.
51. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning transferable visual models from natural language supervision. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021.
52. Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv* **2024**, arXiv:2405.09818.
53. Snell, C.; Lee, J.; Xu, K.; Kumar, A. Scaling LLM test-time compute optimally can be more effective than scaling model parameters for reasoning. *arXiv* **2024**, arXiv:2408.03314.
54. Xu, F.; Hao, Q.; Shao, C.; Zong, Z.; Li, Y.; Wang, J.; Zhang, Y.; Wang, J.; Lan, X.; Gong, J.; et al. Toward large reasoning models: A survey of reinforced reasoning with large language models. *Patterns* **2025**, *6*, 101370. <https://doi.org/10.1016/j.patter.2025.101370>.
55. Colelough, B.C.; Regli, W. Neuro-symbolic AI in 2024: A systematic review. *arXiv* **2025**, arXiv:2501.05435.
56. Pan, J.Z.; Razniewski, S.; Kalo, J.C.; Singhania, S.; Chen, J.; Dietze, S.; Jabeen, H.; Omeliyanenko, J.; Zhang, W.; Lissandrini, M.; et al. Large language models and knowledge graphs: Opportunities and challenges. *Trans. Graph Data Knowl.* **2023**, *1*, 2:1–2:38. <https://doi.org/10.4230/TGDK.1.1.2>.
57. Lundberg, S.M.; Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*; NeurIPS: San Diego, CA, USA, 2017, 30.
58. Chefer, H.; Gur, S.; Wolf, L. Transformer interpretability beyond attention visualization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021.
59. Elhage, N.; Nanda, N.; Olsson, C.; Henighan, T.; Joseph, N.; Mann, B.; Askell, A.; Bai, Y.; Chen, A.; Conerly, T.; et al. A mathematical framework for Transformer circuits. *Transform. Circuits Thread* **2021** <https://transformer-circuits.pub/2021/framework/index.html>. (accessed on 22 July)
60. Ali, A.A.; Zimmerman, I.; Wolf, L. The hidden attention of Mamba models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*; Association for Computational Linguistics: Vienna, Austria, 2025; Volume 1: Long Papers, pp. 1516–1534. <https://doi.org/10.18653/v1/2025.acl-long.76>.
61. Zhang, Z.; Yang, Y.; Dai, Y.; Qu, L.; Xu, Z.; Xu, Y. Federated large language models: Current progress and future directions. *arXiv* **2024**, arXiv:2409.15723.
62. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1. 2023. Available online: <https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf> (accessed on 5 May 2026).
63. European Parliament and Council of the European Union. Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). *Off. J. Eur. Union* 2024/1689, 12.7.2024CELEX Number: 32024R1689, **2024**.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.