ChatGPT

Modern LLM Architecture Comparison (2025 Update)

It has been roughly seven years since the original GPT architecture, and recent open large-language models (LLMs) in 2024–2025 continue to build on transformer foundations with **incremental but important architectural innovations** [1] [2]. In this report, we compare **the architectures of state-of-the-art open LLMs** from Sebastian Raschka's **"Big LLM Architecture Comparison"** (published July 2025) [3] and include **new models introduced in Q3 2025**. We focus on each model's novel architectural components and design choices – such as attention mechanisms, **Mixture-of-Experts (MoE)** layers, normalization placements, positional encodings, etc. – and discuss their use cases or performance focus. A summary table of key architectural features is provided for reference. (We emphasize peer-reviewed papers for citations whenever available, and company technical reports where necessary.)

Architectural Innovations in 2025 LLMs

Before diving into individual models, it's useful to note common **trends in 2025-era LLM architectures**:

- **Mixture-of-Experts (MoE) Layers:** Many top models now employ MoE to scale parameter count without proportionally increasing inference cost. A transformer's feed-forward network is replaced by multiple parallel expert networks, a subset of which "activate" per token [4]. This allows extremely large total parameter counts (hundreds of billions to trillions) but only a fraction (the *activated* experts) are used for any given token, keeping inference efficient [5] [6]. For example, DeepSeek-V3 has 256 experts per MoE layer with 9 experts active (including a *shared* always-active expert) – so out of **671B** total parameters, only **37B** (≈9%) are used per token [7] [8]. MoE's benefit is higher model capacity (more total parameters = more knowledge stored) without commensurate slowdown at runtime [9]. **Shared experts** (an expert active for every token) are another design seen in some MoE implementations (DeepSeek, GLM-4.5, etc.) to handle generic patterns and improve overall performance [10] [11].

- **New Attention Mechanisms:** Transformers traditionally use full *Multi-Head Self-Attention* (MHA), but recent models often replace or augment this for efficiency. **Grouped-Query Attention (GQA)**, first popularized by Llama 2 (2023), reduces memory by having multiple query heads share the same key/value projections [12]. **Multi-Head *Latent* Attention (MLA)** compresses key/value vectors for caching efficiency [13]. Some models use **sliding window (local) attention** in some layers, limiting each token's attention to a local context window instead of the full sequence [14] [15] – this dramatically cuts memory usage for long contexts with minimal impact on performance [16]. In late 2025, Alibaba's Qwen team introduced a novel **"Hybrid" attention** combining a new long-range mechanism called *Gated DeltaNet* with standard (gated) attention in a 3:1 layer ratio [17] [18]. This allowed Qwen3-Next to natively support extremely long contexts (up to 262k tokens) efficiently [19] [20]. We will see each model's choice (MHA vs GQA vs MLA vs sliding windows vs DeltaNet hybrid) in the comparisons below.

- **Normalization Placement:** The positioning of normalization layers (LayerNorm/RMSNorm) is another point of variation. Classic transformers (Vaswani et al. 2017) used **post-norm** (norm after attention/FFN) [21], while GPT-2 and most later models use **pre-norm** (norm before each sub-layer) for more stable training [22]. Interestingly, OLMo-2 (2025) went back to a form of post-norm (with RMSNorm) *inside* the residual blocks, citing improved training stability [23] [24]. Google's Gemma 3 took a hybrid approach, applying **RMSNorm both before and after** the attention module (getting "the best of both worlds") [25] [26]. Some models also add **QK-Norm**, an extra RMSNorm on the *queries and keys* within attention to stabilize training further [27] [28]. We'll highlight these choices per model.

- **Positional Encodings:** Nearly all models now use **RoPE** (Rotary Position Embeddings) or its scaled variants for positional information instead of the old absolute positional embeddings [29]. One interesting outlier is SmolLM3, which explored **NoPE (No Positional Embeddings)** – i.e. *omitting* any explicit positional encoding in some layers [30] [31]. NoPE relies on the causal mask alone to impart sequence order, an idea that showed promise for better length generalization in smaller models [32] (though its benefits at scale remain unclear, so SmolLM3 only applies NoPE sparingly in its architecture [33]).

- **Depth vs Width:** With fixed parameter budget, one can make a model **deeper (more layers)** or **wider (larger layer dimensions)**. Different teams took different approaches. For example, Qwen3 30B vs gpt-oss-20B: Qwen3 is much deeper (48 layers vs 24) whereas OpenAI's GPT-OSS went for width (larger embedding and feed-forward dimensions) [34] [35]. A Gemma 2 ablation found a slight performance edge for wider models at ~9B scale [36], but generally the trade-off remains context-dependent. Deeper models can be harder to train (gradient issues), whereas wider models use more memory but can be faster at inference due to parallelism [37] [38]. We will note such differences (e.g., GPT-OSS's "fatter" layers vs Qwen's deeper stack).

In the sections below, we examine each notable LLM architecture of 2024–Q3 2025, roughly in the order they were introduced. For each, we outline key architectural features and any unique design decisions, and we mention primary intended use cases or notable performance characteristics (e.g. coding ability, reasoning, etc.).

DeepSeek-V3 and DeepSeek-R1 (Open, 2024–25)

DeepSeek-V3 (Dec 2024) is a seminal open LLM architecture that introduced several innovations later adopted by others [39] [40]. It's a **671-billion-parameter** MoE transformer – one of the largest open models of its time – but only **37B parameters are active per token** thanks to its sparse MoE design [41] [8]. The model has **256 experts per MoE layer** (plus 1 shared expert), with **9 experts activated** for each token (8 chosen by a router + the shared expert) [7] [8]. This yields massive capacity without a proportional runtime hit. DeepSeek-V3 also employs **Multi-Head Latent Attention (MLA)** in place of standard attention or GQA [42] [43]. In MLA, the key/value vectors are *compressed* before storing to the KV cache (and reprojected during use) to save memory [13]. Ablations in the DeepSeek papers showed MLA can slightly **outperform** standard attention in modeling quality while greatly reducing memory usage [44] [45] – one reason the DeepSeek team chose MLA over simpler GQA. Aside from MLA and MoE, DeepSeek-V3 uses other modern conventions: SiLU/SwiGLU activations, RMSNorm, RoPE, etc., largely sticking to a standard transformer block design [46].

DeepSeek-R1 (Jan 2025) is a variant fine-tuned specifically as a *"reasoning model"*, i.e. optimized to produce step-by-step chains-of-thought for complex problem solving [47] [48]. Architecturally, R1 is *identical* to V3 [49]; the differences lie in training (R1 underwent reinforcement learning to encourage long-form reasoning, per DeepSeek's technical paper [50] [51]). Notably, R1's release in early 2025 made headlines for achieving **state-of-the-art reasoning, math, and coding performance** on benchmarks – rivaling closed models like OpenAI's "o1" and Anthropic Claude, but at a fraction of the cost [52] [53]. This demonstrated the effectiveness of the DeepSeek-V3 architecture when combined with specialized training. In short, DeepSeek-V3/R1's contributions were **huge model capacity with MoE (671B parameters, 256 experts)** [8] **and a novel MLA attention mechanism** [41], making it an extremely powerful yet *inference-efficient* open model. DeepSeek-R1's success also inaugurated the trend of **"reasoning LLMs"** (models fine-tuned for multi-step reasoning) in the open-source world [54] [55].

Use Cases: DeepSeek-R1 is explicitly geared towards **complex reasoning tasks** – e.g. multi-step math word problems, coding challenges, anything benefited by chain-of-thought [53] [56]. Its strong coding and math abilities are widely noted [57] [58]. With 671B weight, it's resource-intensive to run; however, its MoE design means it's *faster* at inference than dense models of comparable total size [41] [5]. In practice, DeepSeek models require powerful multi-GPU setups or TPU pods, so they are used mainly in research labs or large-scale deployments rather than consumer laptops. Their open release nonetheless made cutting-edge reasoning AI accessible for those with sufficient hardware or via cloud API. Additionally, the DeepSeek architecture served as a **template for later models** (as we'll see, Kimi K2 basically scaled up DeepSeek-V3's design [59], and Llama 4's design is very similar as well [60] [61]).

OLMo 2 (Allen Institute AI, Jan 2025)

OLMo 2 is a family of open models released by the Allen Institute for AI (AI2) as part of their **"Open Language Model"** project. While not the absolute top of leaderboards, OLMo 2 is notable for its **transparency** – detailed technical report, released code, and openly available training data – making it a great learning resource for LLM developers [62] [63]. Architecturally, OLMo 2 models (7B and 13B param dense transformers [64]) follow the basic GPT-style transformer but with two key tweaks for training stability: (1) **Nonstandard normalization placement (Post-Norm)** and (2) **QK-Norm**. Specifically, OLMo 2 uses **RMSNorm layers *after* the attention and feed-forward sub-layers** (inside the residual), instead of the usual pre-layer norm [23] [24]. This is essentially a return to **post-layernorm** (as in the original Transformer) albeit using RMSNorm. The OLMo paper reported that this reordering, combined with QK-norm, led to more stable loss curves during training [65] [66] (the authors had a plot showing improved stability, though it combined both effects) [67]. OLMo 2 also introduced **QK-Norm**, which is an extra RMS normalization applied to the **queries and keys** *inside* the attention mechanism (before applying RoPE) [68] [69]. This trick, originally from a 2023 vision transformer paper, further stabilizes training by preventing extreme query/key activations [70]. In code, QK-norm means before computing attention, they do: `queries = RMSNorm(queries); keys = RMSNorm(keys)` [71] [72]. Together, OLMo's **Post-Norm + QK-Norm** helped it achieve very smooth training dynamics [73] [70]. Otherwise, OLMo 2 uses fairly standard components: *no MoE (it's a dense model)*, standard multi-head attention (MHA) rather than GQA or MLA [74] (though interestingly, AI2 later did release a 32B variant with GQA in mid-2025) [75], and RoPE positional encoding, GeLU activation (in v1) or maybe SwiGLU in v2 (not certain). The **bottom line** is that OLMo 2 prioritized *robustness and openness* over raw SOTA performance.

One of the OLMo-2 paper's claims to fame was an efficiency study: **as of Jan 2025, OLMo-2 13B was on the Pareto frontier of performance vs. compute cost** – meaning it offered excellent performance for its

training compute among open models [76]. (This was before Llama 4, Gemma 3, Qwen 3, etc., came out, which have since shifted the frontier.) OLMo 2 isn't a giant model, but it "punches above its weight" and is very well-documented.

Use Cases: OLMo 2's **7B and 13B** models were positioned as **fully open general-purpose LLMs** – similar in size to LLaMA-1 7B/13B or GPT-3's smaller variants. They can be fine-tuned for chat or used for research. With their open license and transparent training, they're great for academic use or as base models to experiment with novel training techniques. OLMo 2 also provided an example of how to incorporate *stability tricks* (norm tweaks) for others to follow. While their raw performance is moderate (they won't beat a 70B Llama on most tasks), they serve as a **"blueprint"** for building reliable LLMs [62] [77]. AI2's emphasis on documentation means OLMo is often recommended as a teaching tool for understanding LLM training processes.

Gemma 3 (Google, Mar 2025)

Gemma 3 is Google's third-generation open LLM (successor to Gemma 2, which came out in late 2024). It's known to be **highly performant for its size (27B)** and somewhat underappreciated in open-source circles [78] [79]. A distinguishing feature of Gemma: it focuses on a **27B model** (with smaller 1B,4B,12B variants also released), rather than going for 70B or larger. Google found 27B hits a sweet spot of capability vs. efficiency – indeed, Gemma-3 27B runs comfortably on a Mac Mini and significantly outperforms 7–13B models. Architecturally, Gemma 3's headline innovation is its use of **Sliding Window Attention** in most layers to handle long contexts more efficiently [80] [15]. Sliding window (a form of *local attention*) means each token attends only to a fixed-length window of recent tokens instead of the entire sequence [81] [82]. Gemma 3 uses a hybrid: out of every 6 layers, 5 use sliding-window attention and only 1 is full global attention [83]. Moreover, the window size in Gemma 3 is 1,024 tokens, reduced from Gemma 2's 4,096 [83]. This design dramatically reduces KV cache memory – the Gemma 3 paper shows over **5× lower** memory use with minimal perplexity loss [84] [16]. Notably, Gemma 3 applies sliding windows on **grouped-query attention (GQA)** heads (they still moved to GQA like Llama 2) [85] [82]. The team's ablation confirmed that this local attention caused *little to no drop* in model quality [16], validating it as an effective efficiency hack.

Another quirky feature: **Normalization placement in Gemma 3**. It uses *two* RMSNorms around the attention/FFN – one *before* and one *after* (sometimes called **sandwich or pre+post norm**) [25] [86]. This was carried over from Gemma 2. As Raschka notes, it's an intuitive "best of both worlds" approach; the extra norm likely aids training stability and doesn't hurt runtime much (RMSNorm is cheap) [87]. In contrast, Llama and others use only pre-norm, and OLMo 2 did a variant of post-norm – Gemma does both. Aside from that, Gemma 3 is a dense model (no MoE in the main 27B model) and uses typical components: RoPE, SiLU activations, etc. However, Google did release a **Gemma 3n** variant a few months later, optimized for *on-device* use [88]. Gemma 3n introduced **Per-Layer Embedding (PLE)**, which offloads some embeddings to CPU/SSD to save GPU memory [89]. It also implemented **"MatFormer" (Matryoshka Transformer)** slicing – training one model that can be *partially loaded* as smaller models (e.g. use only a subset of layers for a lighter model) [90]. These are advanced efficiency tricks beyond core architecture, aimed at enabling Gemma to run on phones.

Use Cases: Gemma 3 (27B) is a well-rounded model known for strong performance on natural language tasks (and *multilingual* ability, partly due to a larger vocab) [78]. Its efficient attention makes it good for longer inputs (e.g. documents up to, say, 16k tokens or more) without too much memory cost. It's also quite **fast** in practice; one reason being its smaller size relative to 70B models, and possibly because local

attention can use optimized FlashAttention kernels for the global parts. In fact, Gemma 3's design was somewhat **latency-optimized**: by using mostly local attention, it might avoid some quadratic overhead (though local attention still scales linearly with window size). Users have found Gemma 3 to be an **excellent balance** for local deployment – *"it runs just fine on my Mac Mini"*, as Raschka quips, yet it's much more capable than 7B/13B models. Overall, think of Gemma 3 as Google's open competitor to LLaMA, with clever efficiency tweaks. It's suitable for chat assistants, language understanding tasks, etc., especially when one can't host very large models but wants near-state-of-art quality.

(Note: Gemma's under-the-radar status might be due to less hype; but it ranked near top on many benchmarks in 2025. Its one limitation is being 27B – truly hard tasks might still need a 70B+ model. Google's follow-up, Gemma 4, is anticipated to scale this approach further.)

Mistral Small 3.1 (Mistral AI, Mar 2025)

Mistral Small 3.1 is an open 24B-parameter model that made waves by **outperforming larger 30B–40B models** while being *faster* at inference [91] [92]. Released by the startup Mistral AI in March 2025, it was touted as "the best model in its weight class" [93] – indeed, it surpassed Gemma 3 (27B) on many benchmarks (except math) [91]. How did Mistral achieve this? Part of it was **training/data**, but architecturally, *latency optimization* was key. Mistral 3.1 uses a **custom tokenizer** (possibly reducing sequence length overhead) and notably **reduced the number of layers** compared to a typical 24B model, thereby shrinking the KV cache size and speeding up inference [94] [95]. Essentially it trades some depth for speed. The model ended up with a shorter stack and/or narrower layers than, say, Gemma 3 – this yields higher token throughput. Additionally, earlier Mistral versions had experimented with local attention (like Gemma), but **Mistral 3.1 abandoned sliding-window attention and reverted to standard global attention (with GQA)** [96]. This might sound counter-intuitive, but the reasoning is likely that using standard attention lets them leverage highly optimized kernels (FlashAttention, fused ops) end-to-end, whereas custom sliding attention might not be as optimized on all hardware [97]. The Mistral team speculated that while sliding window saves memory, it doesn't necessarily reduce *latency* – and their focus for 3.1 was to minimize latency per token [97]. So by using plain GQA (global attention) and keeping the model relatively shallow, they could run it extremely fast (they reported ~150 tokens/sec generation speed in one setting [98] [99]). Mistral 3.1 also features a **128k context length** – a huge context window enabled presumably by positional interpolation or a segmented attention approach (the details aren't fully described, but the model card notes 128k support) [98]. It's also **multimodal** (trained with image tokens for vision + text) out of the box [100], and supports multiple languages. Importantly, Mistral 3.1 is released under Apache 2.0 license [101], making it very permissive for commercial use.

In summary, Mistral Small 3.1's architecture doesn't introduce novel modules; rather it **optimizes known components for speed**. It uses grouped-query attention (like LLaMA) without fancy sparsity or MoE, and standard RoPE pos. encoding. The standout is its **efficient implementation** and the engineering choices (tokenizer, layer count, kernel optimization) that let a 24B model punch at the level of a 30B in quality while being as fast as perhaps a 13B.

Use Cases: Mistral 3.1 is designed for **real-world deployment where latency and cost matter**. Its creators explicitly mention use in **enterprise applications**, virtual assistants, on-device scenarios, etc., highlighting features like fast response, low resource requirements (runs on a single high-end GPU or even on a Mac with 32GB RAM) [102] [103]. It's versatile: capable in text tasks, vision+text understanding, and very long context processing (e.g. analyzing long documents or multiple documents up to 128k tokens). The model

also supports **function calling** and similar agentic behaviors out-of-the-box [104]. Given its strong performance and permissive license, many community projects have adopted Mistral 3.1 as a base model for fine-tunes, including advanced reasoning agents (the Mistral team noted several "reasoning models" built on top of it by the community) [105]. In short, Mistral 3.1 is a **"fast and free"** alternative to proprietary models, enabling high-quality chat or multi-modal assistants with relatively affordable hardware.

(Note: In late 2025, Mistral AI was expected to release larger models, but 3.1 24B remains notable for showing how far one can push efficiency at moderate scale.)

Llama 4 (Meta AI, Apr 2025)

Llama 4 is Meta's follow-up to the Llama 2/3 series, representing a leap into **Mixture-of-Experts architectures** and massive context lengths. It was first announced in April 2025, with two main variants: **Llama 4 "Maverick" (~400B total, 17B active)** and **Llama 4 "Scout" (~109B total, 17B active)** [106] [107]. Both are MoE models featuring **128 experts (Maverick) or 16 experts (Scout)**, such that only 17B parameters are used per token (roughly comparable to a dense 17B model's compute) [107]. This design lets Meta push the total parameter count to frontier scale (hundreds of billions) while keeping inference economical – *"Activating only ~4.25% of parameters per token"* as one summary notes [108] [109]. Architecturally, Llama 4 is **very similar to DeepSeek-V3** in many respects [60] [61]: it uses MoE layers in an auto-regressive transformer, with modern tricks like SwiGLU, RoPE, etc. However, there are some differences in how MoE is applied. **Llama 4 alternates dense and MoE layers** (every other transformer block is an MoE feed-forward) [110], whereas DeepSeek put MoE in nearly every block (after the first few). Also, Llama 4 uses a more "classic" MoE setup with **fewer, larger experts** – specifically, only **2 experts are active per token**, but each expert's hidden size is very large (e.g. 8192) [111] [112]. In contrast, DeepSeek V3 had 8+1 experts active of size 2048 each [111]. So Llama 4's MoE is *sparser* (fewer experts used) but those experts individually have higher capacity. This means Llama 4 Maverick (400B, 128 experts) ends up using 17B per token, whereas DeepSeek (671B, 256 experts) used 37B per token [113]. Consequently, DeepSeek might have a slight quality edge (more active params), but Llama 4 is lighter to run per inference step. Another difference: **Attention mechanism** – Llama 4 sticks to **Grouped-Query Attention (GQA)** like Llama 2/3 did [114], instead of adopting MLA. MLA is more complex to implement, so Meta likely kept GQA for stability and simplicity. Llama 4 also includes **multimodal support** from the ground up (processing text + images), though that is beyond our text-focused scope [115]. It was trained on up to **40 trillion tokens** and supports an enormous **256k context window** in pretraining, with fine-tuned versions going up to 1M or even 10M context in special cases [116] [117] (using advanced position encoding techniques like "iRoPE" and memory streaming).

In essence, Llama 4 shows that **Meta embraced MoE** to scale beyond the 100B barrier. The "Maverick" 400B model (128 experts) targets maximum capability (intended to be a GPT-4-class model), while "Scout" 109B (16 experts) is an *efficiency-optimized* model that can even run on a single GPU with 4-bit quantization [118] [117]. Notably, there is mention of an even larger *"Llama 4 Behemoth"* (~2 Trillion parameters, 288B active) in the works [119] [120], but as of Q3 2025 that wasn't released publicly. Llama 4 Maverick remained Meta's flagship open model.

Use Cases: Llama 4 models are **frontier-level LLMs** aimed at both research and enterprise. Maverick (~GPT-4 class) excels at a wide range of tasks including complex reasoning, coding, and multimodal understanding. Scout, being smaller, is geared for easier deployment – Meta specifically optimized it for single-GPU use (with **on-the-fly 4-bit quantization**) to encourage wide adoption [116] [118]. Both models have **native multimodal** abilities (so they can handle image inputs in prompts) [116], making them versatile

for AI assistant applications that involve vision (e.g. analyzing diagrams or screenshots). The huge context lengths (up to millions of tokens in fine-tuned versions) open up use cases like feeding entire codebases or large datasets into the model for summarization or analysis [121]. In enterprise settings, Llama 4 is positioned for companies that require on-premises, private LLM solutions with top-notch performance – Meta even released them under a community license that, while not fully open, allows use with certain terms. In summary, Llama 4 brings **MoE efficiency + long-context + multimodality**, making it one of the most advanced open(-ish) models of 2025.

Qwen 3 (Alibaba, May 2025)

Qwen 3 is the third-generation model series from Alibaba's DAMO Academy (the team behind the **Qwen** open models). Qwen-3 solidified Alibaba as a top contender on open LLM leaderboards in 2025 [122] [123]. The Qwen 3 release is notable because it provided a **whole suite of model sizes** *and* both dense and MoE variants. Specifically, Qwen3 has **7 dense models** ranging from a tiny **0.6B** up to 32B parameters, and **2 MoE models**: **30B-A3B** and **235B-A22B** [123] [124]. The "A3B" / "A22B" notation indicates the number of *active* parameters (e.g. 30B total, 3B active; 235B total, 22B active). By offering both dense and MoE, Alibaba gave users flexibility: the dense models are simpler to fine-tune and deploy on any hardware, while the MoE models achieve higher effective capacity for a given inference budget [125] [126]. In terms of architecture, Qwen3 largely follows the LLaMA-style transformer (pre-norm, RoPE, GQA attention, etc.), with the MoE variants adding expert layers. One interesting detail: earlier Qwen models (like Qwen-2.5 MoE) **used a shared expert** in their MoE layers, but **Qwen3's MoE *removed* the shared expert** [127]. The developers noted that when they scaled from 2 active experts to 8 active experts, they didn't see a significant benefit from having a shared expert, and leaving it out made inference and training slightly simpler (no extra always-on capacity to maintain) [128] [129]. This goes against DeepSeek's approach which kept a shared expert for a performance boost [10] [130]; the Qwen team's decision suggests that at 8 experts, balancing and specialization was achievable without a shared unit. Aside from that, Qwen3's architecture included standard enhancements: for instance, it implemented **QK-Norm** in attention like OLMo did (the Qwen3 code and Raschka's from-scratch reimplementation highlight an optional QK-norm flag) [131] [69]. It also uses **Grouped Query Attention (GQA)**, which by 2025 is the norm for large models (and indeed Qwen2 had it as well). The Qwen3-235B model has 128 experts (inferred from A22B active: likely 8 experts active of ~128 total, since 22B active out of 235B is ~9%, matching DeepSeek's 9/256 ratio) – though the exact expert count isn't explicitly stated, it's presumably similar to Llama 4 or DeepSeek scale. Internally, Qwen3 uses **48 transformer layers** in the big models (as mentioned in comparisons) [35] [34], making it a relatively deep model. By contrast, OpenAI's GPT-OSS (120B) had only 24 layers but wider – Raschka shows Qwen3-30B vs GPT-OSS-20B to illustrate this deep vs wide difference [34] [35].

One very small Qwen3 model deserves mention: **Qwen-3 0.6B**, which might be the *smallest current-gen LLM* released [132]. Despite only 600M parameters, it reportedly performs remarkably well for its size, even outperforming older 1-2B models. Raschka himself adopted Qwen-0.6B in place of LLaMA-3 1B for many purposes due to its speed and surprisingly good accuracy [133]. This highlights Qwen team's strengths in efficient scaling – their models are well-optimized across the board.

Use Cases: Qwen3's lineup covers a broad spectrum. The **235B MoE model** is among the best open models for **raw performance**, topping many benchmarks in the 100–300B range (it's *almost* at the level of DeepSeek 671B and Claude 2, but in open weights) [134] [135]. That model is ideal for demanding tasks like difficult QA, coding, complex reasoning, etc., in research or enterprise settings (assuming you have the compute to serve it). The **dense Qwen3 models (7B–32B)** serve as general-purpose backbones for fine-

tuning and deployment where MoE might be overkill. Many fine-tuned chatbots and specialized models in 2025 were built on Qwen-7B or Qwen-14B, for example. The smallest Qwens (0.6B, 1.7B) are extremely fast and good for edge devices or educational/demo uses. Qwen3 models also have strong **multilingual** capabilities and even some **tool-use skills** (the team participated in a NeurIPS LLM efficiency challenge focusing on those) [122] [136]. Overall, Qwen3 is Alibaba's answer to LLaMA 2/3 and Falcon, and it's widely considered *one of the most robust open model series of 2025*. The decision to release both MoE and dense versions set a precedent that others (like Meta with Llama 4) followed, acknowledging that different use cases benefit from different architectures [137] [138].

SmolLM3 (Hugging Face, June 2025)

SmolLM3 is a 3-billion-parameter model created as a proof-of-concept that **small models can still pack a punch** with the right design. It's not as famous as the others here, but Sebastian Raschka included it for its interesting approach [139] [140]. SmolLM3's architecture is *fairly standard transformer* (think GPT-2 style, just scaled to 3B with modern tweaks), **except for one twist: it partially uses "NoPE" – No Positional Embeddings** [140] [30]. The idea of NoPE (from a 2023 paper by Haviv et al.) is to **not inject any explicit positional encoding** into certain layers, relying only on the autoregressive mask to impart sequence order [30] [31]. In theory, a transformer can learn positional cues implicitly since the causal mask defines an order (token *t* can only attend to tokens < *t*) [141] [142]. The NoPE paper found that small models could generalize to longer sequence lengths better without fixed positional features [32]. However, pure NoPE might make training harder for large models, so SmolLM3 uses a hybrid: it **omits positional encoding in every 4th layer** (while using RoPE in others) [33]. This presumably gives it some of the length-generalization benefit without destabilizing the whole network. Apart from NoPE, SmolLM3 uses typical components (GELU or SwiGLU, layernorm, etc.). It's a **dense 3B model** (no MoE). The creators provided extensive training details and comparisons showing SmolLM3 could outperform larger models like LLaMA-3 3B and Qwen-3 4B in certain evaluations [143] [144]. Indeed, Raschka's figure shows SmolLM3 had higher win-rates in pairwise evals versus those models [144]. This goes to show careful optimization and training can make a 3B model competitive with 4B ones.

Use Cases: As the name implies, *"Smol" LM* is attractive for **lightweight deployments** – it's small enough to run on CPU or mobile devices (with quantization). It sits in between 1-2B models (like GPT-2 XL) and the popular 7B+ models. Given its surprisingly good performance, SmolLM3 could power chatbots or assistants on devices where memory is constrained. Also, because the team open-sourced training logs and methodology, it serves as a learning case for **training small efficient LMs** (similar to how OLMo is for larger ones). The NoPE aspect might interest researchers studying positional encoding effects. Length-generalization is a big issue – many models struggle as input length grows beyond training range. NoPE theoretically helps mitigate that [32], so SmolLM3 might handle longer inputs more gracefully than a typical 3B model (though at 3B, absolute performance on very long inputs is limited). In summary, SmolLM3 isn't pushing SOTA, but it's an **"interesting little model"** that shows innovation in position encoding and fills a niche between tiny and mid-sized LLMs.

Kimi K2 (Moonshot AI, Jul 2025)

Kimi K2 (also called *Kimi "2.0" or Kimi 2.5*) is a landmark open model for a few reasons: it is **the first 1-trillion-parameter open-weight LLM** and at release it was arguably *the most capable open model*, reaching parity with top proprietary models like GPT-4, Claude 2, and Google Gemini on many tasks [145] [146]. Developed by Moonshot AI (a Chinese startup), Kimi K2 was open-sourced in July 2025 accompanied by an

arXiv paper [147]. Kimi K2's architecture **directly builds on DeepSeek-V3's design** – in fact, Raschka notes it's essentially *"DeepSeek V3 made larger"* [148]. It retains the **Multi-Head Latent Attention (MLA)** mechanism and MoE structure, but scales some aspects: Kimi K2 has **384 experts** per MoE layer (vs 256 in DeepSeek) and uses *fewer attention heads* (64 heads vs 80 in DeepSeek V3, according to one source) [149] [58]. The MoE routing in Kimi still activates 8 experts + 1 shared (total 9 active) per token, if following DeepSeek's pattern. With **1.04 trillion total params** and **32B activated params per token** [150] [58], Kimi K2 essentially has *half the density* of DeepSeek V3 (which was 37B active out of 671B). Even so, 32B active is huge, and Kimi's training recipe allowed it to **significantly outperform DeepSeek V3/R1** despite the latter's higher active count [151]. A major factor was **training stability**: Kimi K2 pioneered the use of the **Muon optimizer** at scale [152]. Muon is a new optimization algorithm (introduced in 2024) that uses partial orthogonalization of gradients to escape sharp minima. It had only been shown to work up to 16B models before [153], but Moonshot applied it to 1T with great success (they reported *exceptionally smooth* loss curves and no mode collapse) [152] [154]. The Muon optimizer (and an improved variant MuonClip) led to **2× training efficiency and 50% memory reduction**, and enabled training Kimi K2 on 15.5T tokens without divergence [155] [156]. This is a big deal – many earlier attempts at trillion-param training hit instabilities.

The resulting model Kimi K2 has 61 layers, 64 attention heads, and a context window of 128k tokens [58]. It uses a **"linear" RoPE scaling** for the long context (the earlier Kimi-1.5 could handle 2 million char inputs via a special technique – Moonshot values *"lossless long-context"* highly) [157] [158]. Kimi K2 is also oriented towards "agentic" behavior: it was shown autonomously using tools to perform multi-step tasks (e.g. a 17-step plan involving search, calendar and email tools) [159]. This is partly due to fine-tuning – Kimi K2 was trained not just on raw data but also on complex reasoning and tool-use demonstrations (as suggested by it excelling in the *TAU* agent benchmark and Moonshot's demos) [160] [159].

Notably, Kimi K2's *predecessor*, Kimi 1.5 (Jan 2025), was another MoE model (around 314B total) focusing on **reinforcement learning with LLMs** [161]. It was overshadowed by DeepSeek R1 releasing the same day and its weights weren't public. For Kimi K2, Moonshot made a point to open-source the model **before DeepSeek R2 came out** [162]. This competitive dynamic pushed open models forward.

Use Cases: Kimi K2 is a **frontier model** suitable for any high-end NLP task – coding, mathematical reasoning, planning, you name it. It was explicitly designed with **software engineering and "agentic" tasks** in mind [163] [57]. Indeed, it beats GPT-4 on coding benchmarks like LiveCode and is among the best on agent benchmarks [160] [164]. With its huge context, it can take extremely large inputs (e.g. multiple documents, long transcripts). However, with 1T parameters, Kimi K2 is very resource-intensive – serving it likely requires multiple A100/A1000 GPUs even with 4-bit quantization. Thus it's currently used in **research lab settings and by companies with substantial hardware** (or via cloud APIs). Its open availability under a permissive license (Apache 2.0) [165] means it can be used for developing advanced agents, code assistants, or domain-specific expert LLMs. Moonshot's philosophy behind Kimi is striving for **AGI**; they view long context and massive scale as keys to that [158] [166]. So Kimi K2 is both a tool and an experiment in pushing the boundaries of LLM capabilities. As of Q3 2025, it sits at the top of open model performance rankings, slightly above other contenders like GLM-4.5 and Qwen3-Next (discussed next) [167] [168].

(Moonshot's success with Kimi K2 also underscored China's significant presence in LLM research, alongside models like Baichuan, InternLM, and Ziya in 2025. Kimi K2's strong open release has put pressure on others to open-source their best as well.)

OpenAI GPT-OSS (20B & 120B, Aug 2025)

OpenAI surprised the community in August 2025 by releasing **GPT-OSS-20B and GPT-OSS-120B**, their first open-weight LLMs since GPT-2 in 2019 [169] [170]. These models (OSS stands for Open Source Series, or Open-Weight Shared, as some joked) are **"reasoning-optimized"** models intended to deliver strong performance on complex tasks at low cost [170] [171]. Architecturally, GPT-OSS models use a **Mixture-of-Experts Transformer** similar in spirit to Qwen-3 and Llama 4. The 120B model is MoE with presumably ~32 experts (we know it has very few experts – see below), while the 20B may actually be dense (OpenAI didn't explicitly say, but the 20B's "active" params equal total 20B, implying no MoE). The design choices of GPT-OSS can be inferred from Raschka's analysis and OpenAI's model card:

- **Sliding Window Attention:** GPT-OSS-120B uses *local attention in every other layer* (a 50% ratio) [172] [173]. This is similar to Gemma's idea but evenly interleaved instead of 5:1. The context length is 32k by default, but with the sliding windows, effective context can stretch further. This suggests OpenAI prioritized long context handling efficiently – indeed, they mention *"MXFP4 quantization and efficient kernels"* in the release, hinting at optimized long-context support [174].

- **Few Large Experts vs Many Small:** Interestingly, GPT-OSS **has only 32 experts (with 4 active) in the 120B model**, whereas Qwen3-235B had 128 experts (8 active) [175] [176]. This means each GPT-OSS expert is quite large (if 4 active out of 32 total = 12.5% active, whereas Qwen was ~9% active). OpenAI basically chose *fewer, bigger experts*, contrary to the recent trend of more but smaller experts (DeepSeek and others found more experts yields better specialization [177]). OpenAI's approach might simplify routing and reduce network overhead, at the possible cost of a bit of efficiency. Raschka notes this contrast explicitly [175] [176]. Neither GPT-OSS nor Qwen3 use shared experts [178].

- **Width vs Depth:** The GPT-OSS models are **"wider" rather than deeper**. The 120B has only 24 layers (compared to Qwen3-235B's 48) but a larger hidden size (2880 vs 2048 in Qwen) and more heads (perhaps 32 heads vs 16) [34] [179]. The intermediate FFN (expert) dimension is also larger (2880 vs 768 in Qwen's 30B) [179]. This aligns with an ablation they cite (Gemma 2's Table 9) that slightly favored width at equal param count [36]. So GPT-OSS is built to maximize single-layer expressiveness and inference throughput (wider layers can be parallelized more) [37]. Indeed, OpenAI optimized these for *efficient deployment on consumer hardware*, claiming 120B can run on a single 80GB GPU and 20B on a 16GB device [180] [171]. Being wider helps achieve good token throughput.

- **Attention Bias:** A quirky inclusion is that GPT-OSS uses **bias terms in its attention projections** (the old b_q, b_k, b_v in GPT-2, often dropped in newer models) [181]. This was surprising, as those biases are generally seen as redundant (and indeed the *Attention is all you need* paper even had them, but many later models removed them). Raschka highlights a recent paper that proved the key bias is theoretically unnecessary and showed empirically it doesn't matter much [182]. Why did OpenAI include them? Possibly because they started from a GPT-4 architecture lineage that had them, or they found some slight benefit in certain tasks. In any case, it's a minor detail but shows GPT-OSS architecture wasn't purely copying others – they retained some "OpenAI flavor".

Summing up: GPT-OSS-120B is an **MoE model with 32 experts, 4 active, 24 layers, 2880-dim embeddings**, and uses GQA + sliding local attention. GPT-OSS-20B appears to be a dense 20B, likely 24 layers of 2048-dim or so (given similar architecture diagrams). Both were trained with a mix of supervised fine-tuning and reinforcement learning from OpenAI's proprietary model outputs [183] [184], focusing on reasoning and tool

use. Indeed, OpenAI notes near-parity with their internal *o4-mini* model on reasoning benchmarks for the 120B [171]. The models also have a **"Reasoning mode adjustment"** feature: they can adjust the level of reasoning effort dynamically (e.g. decide to think longer if needed), a result of their CoT training [180] [185].

Use Cases: OpenAI's GPT-OSS models are aimed at **developers and organizations who want capable models on their own infrastructure**. With Apache-2.0 license [170], these models can be integrated into products without worrying about API costs or data privacy. The 20B is ideal for **on-device or edge deployment** – running a relatively powerful AI assistant on a laptop or phone (16GB RAM requirement) [171]. The 120B serves as a **ChatGPT-like model** that companies can fine-tune or extend for their domain, running on a single GPU server or modest cluster. Given OpenAI's branding and the models' demonstrated strengths in reasoning, tool use (they specifically mention >90% success in few-shot function calling and strong performance on the Tau toolkit benchmark) [185], many expect GPT-OSS to be used for building AI agents (with tools like web browsing, code execution) in a self-hosted manner. Essentially, OpenAI provided these as **"GPT-4 for everyone's own data center"** (minus some capability, but also minus the black-box). Early adopters reported that GPT-OSS-120B's quality is very high – nearly reaching their *GPT-4.1 (o4)* on core tasks [180]. So, use cases span from advanced chatbots, coding assistants, to research on model alignment (since one can inspect and modify the model). This release was also strategically important: it addressed mounting pressure for OpenAI to open up, and set a precedent for **open release of very large, instruction-tuned models** by a major player.

(Trivia: The name "OSS" led some to joke it's "Open Source Swift", but it indeed stands for Open-Source Series. Internally, OpenAI had codenamed their open models as the "o" series, like o1, o3, o4, etc., but GPT-OSS is now the public branding.)

Grok 2.5 (xAI, Aug 2025)

Grok 2.5 is the model from Elon Musk's new AI startup **xAI**. It is a **270B-parameter** model that was actually xAI's flagship *closed* model in late 2024, but in Aug 2025 Musk decided to **open-source Grok 2.5's weights** to the public [186] [187]. (He announced that Grok-3 would be open-sourced in ~6 months too [187].) This marked an interesting turn, as Grok was originally a proprietary chatbot rival. Now, architecture-wise, **Grok 2.5 is a fairly standard transformer with a sparse MoE twist**. It uses only **8 experts total** in its MoE layers [188] – a small number compared to others – with probably 2 experts activated per token (that was common in earlier MoE like Switch Transformers [189]). This design of "few large experts" is considered an **"older trend"** in MoEs [188], since newer models (DeepSeek, Qwen) prefer many smaller experts for better specialization [190]. Indeed, Grok 2.5's MoE approach is reminiscent of Google's 2021 GLaM (64 experts, 2 active) or Switch (32 experts, 1 active) in spirit. One notable detail: Grok 2.5's architecture diagram (shared by Raschka) shows an **additional always-on feedforward (SwiGLU) module** alongside the experts, effectively acting as a **shared expert** that is always active [191]. The implementation differed slightly (they doubled its intermediate dimension rather than having a separate small expert) but conceptually, Grok 2.5 *does* include a **shared expert path** in each MoE block [191]. This is interesting because Qwen3 chose to omit shared experts in 2025, whereas xAI included one – perhaps since Grok's tech is from 2024 and followed DeepSpeed's MoE paper which advocated for a shared expert for common skills [10] [130]. Other aspects: Grok 2.5 uses all the **standard modern modules** (GQA attention, RoPE, RMSNorm, etc.). It doesn't push context length hugely (likely 8k or 16k context, not millions like newer ones). Essentially, one can think of Grok 2.5 as *"OpenAI GPT-4 level from 2024, now open-sourced"* – it was reported to be trained on Twitter data and excels at conversational tasks Musk wanted (with maybe some edgier humor and facts since Musk touted it as a truth-seeking AI).

Raschka's summary indicates **architecturally Grok 2.5 is not very exotic** – *"fairly standard overall"* – aside from the MoE layer with 8 experts + shared slot [192] [193]. The small expert count might reflect either a design choice or compute constraint xAI had. It's worth noting xAI also has **Grok 4** internally (likely a larger model) that remained closed; Grok 2.5 was last year's model. Still, having a **real production model's weights open** is rare, and it allows the community to study a model that was tuned for actual product deployment.

Use Cases: With 270B parameters (sparsely activated), Grok 2.5 is quite powerful and was basically the brains behind xAI's own chatbot (which they beta tested as a sort of *TruthGPT*). Now open, it can be used for any high-end NLP tasks – it's particularly good at **general knowledge QA and dialogue**. Musk hinted it had some internet training making it witty for conversations. Benchmarks cited by Raschka show Grok 2.5 performed just slightly below OpenAI's very best and xAI's unreleased Grok 4 on many metrics [194] [167]. So it's definitely in the top tier of open models. For researchers, Grok 2.5 offers a glimpse into a *real* production LLM's weights, which is valuable for alignment and safety studies (how does it differ from others in responses?). The open model can also be fine-tuned by developers to create their own chatbots with a Musk-flavored starting point. Since xAI plans to open Grok 3 later, Grok 2.5 might be a stepping stone for the community – something to experiment with before even larger versions come. Practically, running Grok 2.5 will need significant hardware (like 4+ high-memory GPUs), similar to running a 70B dense model but with some extra overhead for MoE.

In short, Grok 2.5's release was an important moment in Q3 2025 for **transparency**. It's a strong model that **bridges 2024 closed tech to 2025 open research**, and developers interested in a multi-modal (it may have some image understanding, unclear) or social-media-tuned AI might find it particularly useful.

GLM-4.5 (Zhipu AI, Aug 2025)

GLM-4.5 is the latest in the GLM series (General Language Model) from Zhipu AI, a Chinese AI company. Released in August 2025, GLM-4.5 is a **Mixture-of-Experts LLM with 355B total parameters and 32B active** [195] [196]. It comes with a smaller sibling **GLM-4.5-Air (106B total, 12B active)** for efficiency [195] [197]. GLM-4.5 is positioned as an **"agentic, reasoning, and coding"** specialist model (hence 4.5 is nicknamed the ARC model) [196]. Its architecture draws inspiration from DeepSeek and Qwen, but with a couple of unique modifications:

- **Dense Layers Before MoE:** GLM-4.5's transformer blocks are structured such that the first **3 layers are standard dense layers** before any MoE layers begin [198] [199]. In other words, rather than applying MoE from layer 1, they start MoE from layer 4 onward. This idea was borrowed from DeepSeek-V3, which also kept the first 3 layers dense (DeepSeek found it improved convergence) [199]. The rationale is that early layers learn low-level features; introducing MoE (which adds routing instability) too early can disrupt that. By keeping initial layers dense, GLM-4.5 ensures a solid base representation before experts kick in [199]. This helps training stability and final performance – the authors explicitly mention it yields *"stable low-level representations before routing decisions shape higher-level processing"* [199].

- **Shared Expert:** Unlike Qwen3, GLM-4.5 **does include a shared expert** in its MoE layers (similar to DeepSeek) [200]. This means each MoE block has one expert that is always active for every token, plus some number of other experts where only a few are chosen per token. The shared expert likely helps handle common patterns across all inputs (as noted in DeepSpeed-MoE research) [130]. Given

GLM-4.5's size (355B, which likely implies maybe 64 experts with 8 active, or 128 with 4 active), having one always-on expert is a reasonable choice to maximize quality (especially since Qwen3-235B did *not* have one and some thought that might slightly hurt its performance on basic tasks [193]).

- **Attention and Optimizer:** GLM-4.5 uses **Grouped-Query Attention (GQA)** like most big models, and also retains an *Attention bias* term (the *b* vectors in Q,K,V), similar to GPT-OSS, as Raschka noted [201]. It additionally uses the **Muon optimizer** for training (Zhipu mentioned faster convergence due to Muon) [202], indicating cross-pollination of techniques (Muon originated from Moonshot's Kimi research). GLM-4.5 also includes **QK-Norm** in its attention layers, since Raschka listed it as a feature along with GQA [203]. The model has **96 attention heads** – quite high, implying large width or many subheads (InfoQ says it "prioritizes depth over width" but also has 96 heads, which is interesting) [204] [202]. Possibly it's both deep (maybe 64 layers) and still fairly wide (embedding ~2304 or so to accommodate 96 heads of 24-dim each). It also employs **Multi-Token Prediction (MTP)**, which is a training strategy to predict multiple tokens in a single forward pass (kind of like a latent alignment prediction to speed up training) [203]. This helps it handle *"dual modes"* – GLM-4.5 has a special ability to switch between a "thinking" mode (chain-of-thought, tool-using, slow reasoning) and a "non-thinking" fast mode for straightforward prompts [205]. Essentially, it can decide when to engage complex reasoning vs when to just answer directly, optimizing both accuracy and speed.

Performance-wise, GLM-4.5 is **very strong**: on release, Zhipu reported it *outperforms Claude 4 Opus on average across 12 benchmarks*, and is just behind OpenAI's top models and xAI's unreleased Grok-4 [195] [168]. The smaller GLM-4.5-Air (106B) achieves almost the same performance with ~1/3 the total parameters, which is impressive [167]. Figures show GLM-4.5 especially excels in coding tasks (leading many open models on code completion and problem-solving) [206] [207], and has very high tool-use success rates (90.6%, beating even Kimi K2 in one comparison) [206] [207]. This aligns with its "agentic" orientation – it's tuned for being an AI agent that can plan and invoke tools.

Use Cases: GLM-4.5 is a top-tier model for **complex reasoning, coding, and agent applications**. It's essentially China's answer to GPT-4, optimized for tasks requiring multi-step thought and external tool usage (APIs, function calls). For example, GLM-4.5 would be ideal to power an **AI developer assistant** that writes and debugs code, since it's state-of-art on code benchmarks [206]. It's also great for building **autonomous agents** (e.g. an AI that reads documents, then uses a calculator or search engine, etc.) given its high tool-use success. Zhipu offers it via their API (Z.ai platform) and open-sourced the weights for research and fine-tuning [208]. Because of its dual "fast vs think" mode, one might use GLM-4.5 in scenarios where quick responses are needed for easy queries but deep reasoning should trigger for hard queries – possibly making it more efficient in production. The open availability of both the full and "Air" version allows wider adoption: someone with less compute can choose the 106B model and still get near SOTA results. In Chinese contexts, GLM-4.5 is particularly noted to be strong (GLM models historically excel in Chinese tasks as well as English), making it a go-to model for bilingual or multilingual AI systems in Asia. Overall, GLM-4.5 represents the cutting-edge in open LLM design by late 2025, combining lessons from prior models (DeepSeek's MoE strategy, Moonshot's optimizer, OpenAI's CoT techniques) into a single package.

Qwen3-Next (Alibaba, Sept 2025)

Qwen3-Next-80B-A3B was introduced by Alibaba in September 2025 as an **upgrade over Qwen3**, focusing on *efficiency at scale*. It features an **80B total parameter MoE model with only 3B active parameters** –

that is an incredibly high sparsity (just 3.75% of parameters used per token) [209]. To achieve this, Qwen3-Next greatly increased the expert count and reduced expert size: it has **512 experts with 10 active per token** [210] [211] (which likely includes 9 selected + 1 shared expert, as Raschka noted they added a shared expert in Next [212]). Compare this to original Qwen3's 235B which had 8 active out of perhaps 128 experts. Qwen3-Next basically quadrupled the number of experts and cut each expert's size by about 4×, aligning with the trend toward **"many small experts"** for better specialization [188]. The other major innovation is Qwen3-Next's **Hybrid Attention mechanism** for ultra-long contexts. Instead of standard self-attention in each transformer block, it **mixes two types of layers: Gated Attention and Gated *DeltaNet* layers** in a 1:3 ratio [17] [18]. **Gated Attention** is basically scaled-dot-product attention with some gating on the outputs (a per-channel sigmoid gate on the attention output, plus using a *zero-centered* RMSNorm for QK norm, and only applying RoPE on part of the dimensions) [213] [214]. **Gated DeltaNet** is a new long-range sequence model block – likely a variant of the DeltaNet architecture (which is related to state-space models or compressed Fourier features) that allows very long context processing with sublinear scaling. By alternating 1 attention layer for every 3 DeltaNet layers [18], Qwen3-Next achieves a native context length of **262,144 tokens (256k)** [215] [216], far beyond the 32k of Qwen3. And crucially, the memory and compute grow much slower with sequence length in DeltaNet layers, enabling **~10× faster throughput on long sequences** compared to conventional attention models [209]. Essentially, Qwen3-Next is purpose-built for tasks that need to handle *huge* contexts (like processing books, code repositories, or multi-document corpora) efficiently.

Other features: Qwen3-Next uses **Multi-Token Prediction** during pre-training (predicting multiple tokens in one go), which improves training speed and might help with throughput during inference as well [217]. It's also tuned for **advanced reasoning, coding, creative writing, and agent tasks** – basically a general high-end model like its predecessor, but more efficient and long-context savvy [218] [219]. Alibaba released both an **"Instruct" version and a "Thinking" version** of Qwen3-Next. The instruct version is aligned for user-facing interactions, while the thinking version presumably is optimized for chain-of-thought generation (this mirrors how they had Qwen-3 models and a Qwen-Thought variant earlier).

Use Cases: Qwen3-Next is ideal for **long-context tasks**. For example, if one needs to feed an entire novel or a massive log file into an LLM to analyze, Qwen3-Next can handle 256k tokens natively – that's an order of magnitude beyond most models. And thanks to the DeltaNet layers, it can do so without slowdowns that plague normal attention (its design specifically targets *efficient inference on long inputs* [209]). This makes it great for enterprise applications like processing lengthy financial reports, doing code analysis across a whole codebase, or supporting very long conversations in chat. Its MoE architecture with 3B active means it's extremely *compute-efficient* for its total capacity – essentially you get a model that was trained as an 80B (so it "knows" a lot), but you pay inference cost of ~3B. That's smaller than even LLaMA-2-7B in active size, which is remarkable. Therefore, Qwen3-Next can be deployed relatively cheaply compared to models of similar capability. It also still shines on standard benchmarks (matching Qwen3-235B performance in many cases) [220] [221], so it's a solid general-purpose model too. Given its focus on reasoning and coding, one could use it for building **AI agents (with tools)** that maintain a very long memory (imagine an assistant that remembers everything you ever said in a months-long conversation), or for tasks like **multi-document question answering** (where you feed a whole set of documents and ask questions). Essentially, Qwen3-Next represents the cutting edge in **"efficient large-scale LLM"** – a glimpse of how future LLMs might all operate (massive expert counts, specialized long-context architectures).

Alibaba open-sourced Qwen3-Next under a permissive license like its predecessor, and provides it via API as well. It shows their commitment to pushing open LLM tech forward. With Qwen3-Next, they demonstrated

one path to **extreme context lengths and extreme sparsity** without sacrificing ability. Users who need those features will find Qwen3-Next unrivaled as of Q3 2025.

Comparative Overview of Architectures

To summarize the **common architectural components** of these models, the table below highlights key features of each:

Model (Org, Release)	**Param Count** (Total / Active)	**Architecture Type**	**MoE Experts** (total, active, shared?)	**Attention Mechanism**	**Normalization**	**Positional Encoding**	**Conte Lengt**
DeepSeek-V3/R1 (High-Flyer AI, 12/2024 & 1/2025) [222] [8]	**671B / 37B** active per token	Sparse MoE Transformer	256 experts per layer, **9 active** (8 + **1 shared**) [8]	**Multi-Head Latent Attn (MLA)** [41] (memory-compressed KV)	Pre-Norm (RMSNorm), with **Post-Norm variant in OLMo**	RoPE	8k (R1 tuned CoT)
OLMo-2 (Allen AI, 1/2025) [224]	7B & 13B (dense)	Standard Transformer (dense)	n/a (no MoE)	**Multi-Head Attn (MHA)** (later 32B had GQA) [75]	**Post-Norm** RMSNorm (after sublayers) [225] [24] + **QK-Norm** in attn [27]	RoPE	4k

---

Model (Org, Release)	**Param Count** (Total / Active)	**Architecture Type**	**MoE Experts** (total, active, shared?)	**Attention Mechanism**	**Normalization**	**Positional Encoding**	**Contex Length**
Gemma-3 (Google, 3/2025) [79]	27B (plus 1B, 4B,12B variants)	Standard Transformer (dense)	n/a (no MoE)	**Grouped-Query Attn** + **Sliding Window** (5 local : 1 global layers, 1k window) [83]	Pre- & **Post-Norm** (RMSNorm on both ends of sublayers) [25] [86]	RoPE	4k (wi 1k)
Mistral-3.1 (Mistral AI, 3/2025) [91]	24B (base & instruct)	Standard Transformer (dense)	n/a (no MoE)	**Grouped-Query Attn** (dropped sliding attn for speed) [96]	Pre-Norm (RMSNorm)	RoPE	128k (vision [98] [22]

Model (Org, Release)	**Param Count** (Total / Active)	**Architecture Type**	**MoE Experts** (total, active, shared?)	**Attention Mechanism**	**Normalization**	**Positional Encoding**	**Contex Length**
Llama-4 "Maverick" (Meta, 4/2025) [60] [107]	**400B / 17B** active (128 experts) [107]	Sparse MoE Transformer	128 experts, **2 active** (no shared) [111] (alternating MoE layers)	**Grouped-Query Attn** (every layer) [113]	Pre-Norm (RMSNorm)	RoPE (early fusion for multimodality) [116]	256k pretrai to **1M** tuned
Qwen-3 (Alibaba, 5/2025) [123] [127]	Dense: 0.6B–32B; MoE: **30B/3B**, **235B/22B** active [123]	Dense & Sparse (two versions)	235B MoE: ~128 experts, **8 active** (no shared) [127] [229]; 30B MoE: fewer experts (exact not public)	**Grouped-Query Attn** (dense & MoE)	Pre-Norm (RMSNorm); no shared expert in MoE (for simplicity) [127] [128]	RoPE	8k (32k YaRN)

Model (Org, Release)	**Param Count** (Total / Active)	**Architecture Type**	**MoE Experts** (total, active, shared?)	**Attention Mechanism**	**Normalization**	**Positional Encoding**	**Conte Lengt**
SmolLM-3 (HuggingFace, 6/2025) [143]	3B (dense)	Standard Transformer (dense)	n/a (no MoE)	**Multi-Head Attn (MHA)**	Pre-Norm (LayerNorm/ RMSNorm)	**NoPE** (No Pos. Embeddings) in 1/4 of layers [33]; RoPE elsewhere	2k
Kimi K2 (Moonshot, 7/2025) [230] [58]	**1.04T / 32B** active (384 experts) [150] [58]	Sparse MoE Transformer	384 experts, **9 active** (8 + **1 shared**) [149] [58]	**Multi-Head Latent Attn (MLA)** [231]	Pre-Norm (RMSNorm)	RoPE (128k context)	128k

Model (Org, Release)	**Param Count** (Total / Active)	**Architecture Type**	**MoE Experts** (total, active, shared?)	**Attention Mechanism**	**Normalization**	**Positional Encoding**	**Conte Lengt**
GPT-OSS-120B (OpenAI, 8/2025) [176] [180]	**120B / ~3.6B** active (32 experts) [232] [176]	Sparse MoE Transformer	32 experts, **4 active** (no shared) [176]	**Grouped-Query Attn** + **Sliding Window** (every 2nd layer local) [172]	Pre-Norm (RMSNorm); has attention biases [181] [182]	RoPE	32k (sl windo
Grok 2.5 (xAI, 8/2025) [186] [188]	**270B** (est. ~86B active if 2/8 experts)	Sparse MoE Transformer	8 experts, **2 active** (effectively with **shared expert**) [188] [191]	**Grouped-Query Attn** (like GPT-3 style)	Pre-Norm (RMSNorm)	RoPE	8k

Model (Org, Release)	**Param Count** (Total / Active)	**Architecture Type**	**MoE Experts** (total, active, shared?)	**Attention Mechanism**	**Normalization**	**Positional Encoding**	**Conte Lengt**
GLM-4.5 (Zhipu AI, 8/2025) [195] [204]	**355B / 32B** active (MoE) [196] [204]	Sparse MoE Transformer	(Not disclosed; likely ~64 experts, ~8 active + **1 shared**) [200]	**Grouped-Query Attn**; 96 heads; uses Muon opt. [203]	Pre-Norm (RMSNorm); **first 3 layers dense** (then MoE) [198] [199]; **shared expert** yes [200]	RoPE (maybe with bias)	64k (suppo long ta

Model (Org, Release)	Param Count (Total / Active)	Architecture Type	MoE Experts (total, active, shared?)	Attention Mechanism	Normalization	Positional Encoding	Conte Lengt
Qwen3-Next (Alibaba, 9/2025) [209] [210]	**80B / 3B** active (512 experts) [210]	Sparse MoE Transformer	512 experts, **10 active** (9 + **1 shared**) [212] [210]	**Hybrid:** 1× **Gated Attn** + 3× **Gated DeltaNet** layers [17] [18]	Pre-Norm (RMSNorm); uses QK-Norm (likely)	RoPE + advanced (DeltaNet)	**262k** tokens native [235]

Table: Key architectural features of novel 2025 LLMs. *(#Experts refers to MoE expert count per layer. GQA = Grouped-Query Attention, MLA = Multi-Head Latent Attention, MHA = Multi-Head Attention.)*

References:

- DeepSeek V3 & R1 – *"DeepSeek-V3, the backbone of R1, is a 671B parameter MoE model (37B activated)"* [236] [237] ; *MLA vs GQA performance* [222] .
- OLMo 2 – *Post-norm RMSNorm and QK-Norm stabilize training* [65] [70] .
- Gemma 3 – *Sliding window attn (1k local) yields big memory savings with negligible perplexity hit* [84] [16] .
- Mistral 3.1 – *Outperforms Gemma 3 27B while being faster, via fewer layers & optimized kernels* [91] [96] .
- Llama 4 – *400B MoE (128 experts) with 17B active; 2 experts per token vs DeepSeek's 9* [113] [111] ; *1M+ token context with special RoPE (iRoPE)* [121] .
- Qwen3 – *235B-A22B model (no shared expert), deeper vs GPT-OSS's wider design* [34] [127] .
- SmolLM3 – *No positional embedding (NoPE) in some layers improves length generalization* [32] [33] .

- Kimi K2 – *1T param, 384 experts, 32B active; trained with Muon optimizer enabling stable trillion-scale run* [58] [154] ; *tops GPT-4 in coding (53.7% vs 44.7% on benchmark)* [160] .
- GPT-OSS – *OpenAI 120B MoE, 32 experts (4 active), sliding window every other layer* [172] [176] ; *Apache-2 license, near GPT-4 quality on reasoning* [171] .
- Grok 2.5 – *xAI's 270B model, open-sourced; 8 experts (2 active), with an always-on path* [188] [191] .
- GLM-4.5 – *355B/32B MoE, outperforms Claude 4 on avg, uses DeepSeek's dense-then-MoE strategy* [167] [199] ; *MuON optimizer used for fast convergence* [203] .
- Qwen3-Next – *80B/3B MoE (512 experts, 10 active) with hybrid Gated Attention/DeltaNet enabling 262k context, 10× throughput on long input* [209] [210] .

---

[1] [3] [4] [5] [6] [7] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [47] [59] [60] [61] [62] [63] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90] [91] [92] [94] [95] [96] [97] [110] [111] [112] [113] [114] [115] [122] [123] [124] [125] [126] [127] [128] [129] [130] [131] [132] [133] [136] [137] [138] [139] [140] [141] [142] [143] [144] [145] [146] [148] [149] [152] [153] [161] [162] [167] [169] [172] [173] [175] [176] [177] [178] [179] [181] [182] [186] [188] [190] [191] [192] [193] [194] [195] [198] [199] [200] [201] [212] [213] [214] [222] [224] [225] [226] [229] [230] [231] [232] [233] [234] The Big LLM Architecture Comparison
https://magazine.sebastianraschka.com/p/the-big-llm-architecture-comparison

[2] [134] [135] [174] [209] September | 2025 | Radical Data Science
https://radicaldatascience.wordpress.com/2025/09/

[8] [46] [49] [189] DeepSeek v3 and R1 Model Architecture: Why it's powerful and economical
https://fireworks.ai/blog/deepseek-model-architecture

[19] [20] [210] [211] [215] [216] [217] [218] [219] [220] [221] [235] qwen/qwen3-next-80b • LM Studio
https://lmstudio.ai/models/qwen/qwen3-next-80b

[48] [50] [51] [52] [53] [54] [55] [56] [223] [236] DeepSeek: sorting through the hype | IBM
https://www.ibm.com/think/topics/deepseek

[57] [58] [150] [151] [154] [155] [156] [157] [158] [159] [160] [164] [166] 5 Things You Need to Know About Moonshot AI and Kimi K2, the New #1 model on the Hub
https://huggingface.co/blog/fdaudens/moonshot-ai-kimi-k2-explained

[64] OLMo 2: The best fully open language model to date - Ai2
https://allenai.org/blog/olmo2

[93] [98] [99] [100] [101] [102] [103] [104] [105] [227] Mistral Small 3.1 | Mistral AI
https://mistral.ai/news/mistral-small-3-1

[106] [107] [116] [117] [118] [228] Welcome Llama 4 Maverick & Scout on Hugging Face
https://huggingface.co/blog/llama4-release

[108] [109] [119] [120] [121] Llama 4 and Llama 3 Models | Together AI
https://www.together.ai/llama

[147] [165] [2507.20534] Kimi K2: Open Agentic Intelligence - arXiv
https://arxiv.org/abs/2507.20534

[163] Kimi K2: The 1-Trillion Parameter Open-Source Titan Rewriting the ...
https://medium.com/towardsdev/kimi-k2-the-1-trillion-parameter-open-source-titan-rewriting-the-rules-of-ai-agents-86de16465eb6

168 202 203 204 205 206 207 GLM-4.5 Launches with Strong Reasoning, Coding, and Agentic Capabilities - InfoQ
https://www.infoq.com/news/2025/08/glm-4-5/

170 171 183 184 185 Introducing gpt-oss | OpenAI
https://openai.com/index/introducing-gpt-oss/

180 OpenAI's new open weight (Apache 2) models are really good
https://simonwillison.net/2025/Aug/5/gpt-oss/

187 Musk says xAI open sources Grok 2.5 | Reuters
https://www.reuters.com/technology/musk-says-xai-open-sources-grok-25-2025-08-23/

196 [2508.06471] GLM-4.5: Agentic, Reasoning, and Coding (ARC) Foundation Models
https://arxiv.org/abs/2508.06471

197 GLM 4.5 : The best Open-Source AI model, beats Kimi-K2, Qwen3
https://medium.com/data-science-in-your-pocket/glm-4-5-the-best-open-source-ai-model-beats-kimi-k2-qwen3-b56a5df2ec34

208 zai-org/GLM-4.5 - Hugging Face
https://huggingface.co/zai-org/GLM-4.5

237 DeepSeek-V3 Technical Report - arXiv
https://arxiv.org/html/2412.19437v1