DeepSeek-V3 Architecture

Deep Dive into DeepSeek — Chapter 5

Yin Yang

Foundation Books Project

The one rule of this chapter

The 671B headline gives scale.
It does not give the computation.

- DeepSeek-V3 is a decoder-only Mixture-of-Experts model whose repeated block combines MLA attention with a dense-to-sparse feed-forward schedule.
- Every number in this chapter comes from a released configuration field, a class in the reference implementation, or a documented weight-layout convention — never from the parameter count alone.

The one rule of this chapter

The 671B headline gives scale.
It does not give the computation.

- DeepSeek-V3 is a decoder-only Mixture-of-Experts model whose repeated block combines MLA attention with a dense-to-sparse feed-forward schedule.
- Every number in this chapter comes from a released configuration field, a class in the reference implementation, or a documented weight-layout convention — never from the parameter count alone.

Goal: reconstruct one V3 block, substitute the released dimensions into the MLA and DeepSeekMoE ledgers built in earlier chapters, and reproduce the 671B/37B split by hand.

Roadmap

1. V3 as a model object

Roadmap

1. V3 as a model object
2. The repeated V3 block

Roadmap

1. V3 as a model object
2. The repeated V3 block
3. MLA and DeepSeekMoE, instantiated

Roadmap

1. V3 as a model object
2. The repeated V3 block
3. MLA and DeepSeekMoE, instantiated
4. Multi-token prediction

Roadmap

1. V3 as a model object
2. The repeated V3 block
3. MLA and DeepSeekMoE, instantiated
4. Multi-token prediction
5. Tokenizer, V3.1, and the dense-baseline comparison

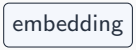
Roadmap

1. V3 as a model object
2. The repeated V3 block
3. MLA and DeepSeekMoE, instantiated
4. Multi-token prediction
5. Tokenizer, V3.1, and the dense-baseline comparison
6. The V3 shape ledger

V3 as a model object

One sentence, one arc, one notation

A decoder-only MoE model whose block combines MLA attention with a dense-to-sparse feed-forward schedule.



One sentence, one arc, one notation

A decoder-only MoE model whose block combines MLA attention with a dense-to-sparse feed-forward schedule.



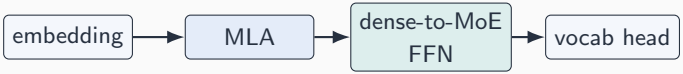
One sentence, one arc, one notation

A decoder-only MoE model whose block combines MLA attention with a dense-to-sparse feed-forward schedule.



One sentence, one arc, one notation

A decoder-only MoE model whose block combines MLA attention with a dense-to-sparse feed-forward schedule.



One sentence, one arc, one notation

A decoder-only MoE model whose block combines MLA attention with a dense-to-sparse feed-forward schedule.



One sentence, one arc, one notation

A decoder-only MoE model whose block combines MLA attention with a dense-to-sparse feed-forward schedule.



The architecture stays the main object here: config and weight files fix dimensions, the repository demo shows one inspectable path, release notes describe public service behavior — each source keeps the question it actually answers.

The repeated V3 block

Pre-norm, residual-preserving, dense-to-sparse

$$H'_l = H_l + \mathrm{MLA}(\mathrm{RMSNorm}(H_l)), \qquad H_{l+1} = H'_l + \mathrm{FFN}_l(\mathrm{RMSNorm}(H'_l)).$$

Block dimension	Dense baseline	V3
Attention cache	full per-head K, V	latent KV + separate RoPE key
Feed-forward path	same dense MLP always	3 dense layers, then shared+routed MoE

`n_layers`=61, `n_dense_layers`=3: layer l is dense MLP if $l < 3$, else shared-plus-routed MoE. The residual-state shape $T \times d$ survives every addition unchanged.

MLA and DeepSeekMoE, instantiated

MLA: 576 cached scalars, not 40,960

$$r_{\mathrm{kv}} = 512, \quad d_{\mathrm{rope}} = 64 \;\Rightarrow\; 512 + 64 = 576 \text{ cached scalars/token.}$$

$$128 \cdot (192 + 128) = 40{,}960 \text{ scalars/token, dense full-KV baseline.}$$

- Query side: `q_lora_rank`=1536 → 128 heads, 192 query-key coordinates each — never cached.

MLA: 576 cached scalars, not 40,960

$$r_{\mathrm{kv}} = 512, \quad d_{\mathrm{rope}} = 64 \;\Rightarrow\; 512 + 64 = 576 \text{ cached scalars/token.}$$

$$128 \cdot (192 + 128) = 40{,}960 \text{ scalars/token, dense full-KV baseline.}$$

- Query side: `q_lora_rank`=1536 → 128 heads, 192 query-key coordinates each — never cached.
- The absorbed inference path folds W^{UK} into the query side, so the per-head key view is never materialized.

MLA: 576 cached scalars, not 40,960

$$r_{\mathrm{kv}} = 512, \quad d_{\mathrm{rope}} = 64 \;\Rightarrow\; 512 + 64 = 576 \text{ cached scalars/token.}$$

$$128 \cdot (192 + 128) = 40{,}960 \text{ scalars/token, dense full-KV baseline.}$$

- Query side: `q_lora_rank`=1536 → 128 heads, 192 query-key coordinates each — never cached.
- The absorbed inference path folds W^{UK} into the query side, so the per-head key view is never materialized.

40,960 ÷ 576 = 71.1× fewer stored scalars — a dimension ratio; latency and memory traffic still depend on the implementation.

DeepSeekMoE: 256 stored, 8 active

$$\texttt{n_routed_experts} = 256, \quad \texttt{num_experts_per_tok} = 8, \quad \texttt{n_shared_experts} = 1.$$

- Group-limited routing: 8 expert groups narrow to 4 candidate groups *before* the top-8 experts are chosen; sigmoid scoring, `route_scale`=2.5.

DeepSeekMoE: 256 stored, 8 active

$$\texttt{n_routed_experts} = 256, \quad \texttt{num_experts_per_tok} = 8, \quad \texttt{n_shared_experts} = 1.$$

- Group-limited routing: 8 expert groups narrow to 4 candidate groups *before* the top-8 experts are chosen; sigmoid scoring, `route_scale`=2.5.
- Token output = weighted routed-expert sum + one shared-expert output — both terms land in the same hidden dimension.

DeepSeekMoE: 256 stored, 8 active

$$\texttt{n_routed_experts} = 256, \quad \texttt{num_experts_per_tok} = 8, \quad \texttt{n_shared_experts} = 1.$$

- Group-limited routing: 8 expert groups narrow to 4 candidate groups *before* the top-8 experts are chosen; sigmoid scoring, `route_scale`=2.5.
- Token output = weighted routed-expert sum + one shared-expert output — both terms land in the same hidden dimension.

671B main-model total, 36.7B (≈37B) active: compare only within one named, matched scope — never total against a different scope's active figure.

Multi-token prediction

From concept to code: MTP is documented, not executed

The pinned reference implementation runs only the 61-layer main path; the auxiliary module is described in the weight documentation instead.

from `DeepSeek-V3/README_WEIGHTS.md`

```
Main Model Weights: model.layers.0 to model.layers.60
MTP Modules: model.layers.61.self_attn & mlp
Loading Rules: MTP layer ID appended immediately
  after the Main Model hidden layers.
```

From concept to code: MTP is documented, not executed

The pinned reference implementation runs only the 61-layer main path; the auxiliary module is described in the weight documentation instead.

from `DeepSeek-V3/README_WEIGHTS.md`

```
Main Model Weights: model.layers.0 to model.layers.60
MTP Modules: model.layers.61.self_attn & mlp
Loading Rules: MTP layer ID appended immediately
  after the Main Model hidden layers.
```

`num_nextn_predict_layers`=1. A checkpoint tensor is not a serving guarantee: whether a runtime repurposes MTP for speculative decoding is a separate, runtime-specific fact.

[INSPECTED: DeepSeek-V3 weight documentation]

Tokenizer, V3.1, and the dense-baseline comparison

Four owners: architecture, tokenizer, template, runtime

- `LlamaTokenizerFast`, BOS=0, EOS=1, `model_max_length`=131072 — distinct from `max_position_embeddings`=163840. Two bounds, two sources.
- V3.1's model card retains V3's structure (671B/≈37B/128K) while adding two-phase long-context training, reported reasoning/agent gains, and UE8M0 FP8 scale metadata.

UE8M0 names a scale format, not the payload — shape compatibility is not the same claim as reproducing the artifact.

Dense baseline, priced dimensionally

$$3dm = 396{,}361{,}728 \text{ (dense FFN)}, \qquad 8 \cdot 3dm_e = 352{,}321{,}536 \text{ (V3, one token)}.$$

- Sparse routing changes *which* matrices a token uses; the checkpoint still stores every routed expert regardless.
- No single "the" dense equivalent exists: matching width/depth, active parameters, or total parameters each fails to match something else.

This is a dimension-based account. Benchmark numbers, latency, and service rate require a dedicated measurement protocol this chapter does not supply.

The V3 shape ledger

Three ledgers, three questions

Ledger	Question
671.026B	checkpoint capacity (main-model scope)

Three ledgers, three questions

Ledger	Question
671.026B	checkpoint capacity (main-model scope)
37.552B mechanical	one token's reference-model path

Three ledgers, three questions

Ledger	Question
671.026B	checkpoint capacity (main-model scope)
37.552B mechanical	one token's reference-model path
42.906B at $W{=}16$	rank-local ownership under one partition

Three ledgers, three questions

Ledger	Question
671.026B	checkpoint capacity (main-model scope)
37.552B mechanical	one token's reference-model path
42.906B at $W{=}16$	rank-local ownership under one partition

All three follow from the same source-pinned dimensions: hidden width, layer and expert counts, MLA ranks. The mechanical 37.552B and the source-reported 36.7B differ by 0.852B and stay labeled as such — neither overwrites the other.

Concept-to-code map for the chapter

- **model object as a dataclass** → `DeepSeek-V3/inference/model.py`
- **dense/MoE cutover, per layer** → `DeepSeek-V3/inference/model.py`
- **MLA cache buffers, absorbed path** → `DeepSeek-V3/inference/model.py`
- **group-limited routing, combine** → `DeepSeek-V3/inference/model.py`
- **MTP module layout** → `DeepSeek-V3/README_WEIGHTS.md`
- **chat template invocation** → `DeepSeek-V3/inference/generate.py`
- **quantized linear dispatch** → `DeepSeek-V3/inference/kernel.py`
- **the shape ledger, runnable** → `code/chapter_deepseek_v3_architecture/`

One block, substituted with real numbers, eight times over.

Next: the pretraining chapter follows the training state that produces this fixed architecture object; the serving chapter (Volume II) follows its downstream execution.