Where Computation Comes From

The Transformer Read as a Net, and the Two Assumptions It Still Carries

J. Konstapel — Constable Research, Leiden 19 September 2026

All Rights Reserved.

1. The Claim

This essay makes one claim and derives it in full.

The claim: the machinery of modern artificial intelligence has been converging, step by step and without a theory, on the operations of the vacuum net. Four of those operations are already inside the code that runs today's largest language models. Two assumptions of the old paradigm remain. Both can be removed. The removal is not a hope. Each removal has already been measured, separately, at scale, by independent groups. What has not been done is the combination, and the derivation that shows why the field moves this way.

This essay supplies the derivation. It reads one concrete piece of code — the published inference file of DeepSeek-V3 — as evidence. It states four theses and derives each one with numbers. It reports what has been measured, in joules, gigabytes and milliseconds. It treats the one serious counter-measurement in the literature in its own section. It ends with the proof agenda: what is derived, what is measured, and what remains open.

Everything needed to follow the argument is in this text. No reference has to be retrieved to check a step.

2. The Net, In Brief

The vacuum net is a fishing net.

Picture the net a fisherman spreads on the quay. It has strands. The strands wind around each other. Where windings close, there is a knot. Between the knots there are meshes. The whole carries tension. Pull anywhere and the tension redistributes everywhere. Nothing in the net is a thing apart from the net. A knot is not an object placed on the strands. A knot is what the strands do at that place.

Five terms carry the whole theory. **Strand**: the one continuous line. **Winding**: a turn of the strand around a reference. **Closure**: a winding that returns in phase with itself. **Mesh**: the opening held between closures. **Tension**: the state of the whole, redistributed instantly, never located.

Two consequences follow directly.

First: there is no background. There is no stage on which the net lies. The net is all there is. What physics calls the vacuum is the net at rest. What physics calls a particle is a closure. What physics calls a constant is a function of the state of the net, not a fixed number written on the stage.

Second: description replaces explanation. Explanation is causal thinking. It needs two separate things and an arrow between them. In a one-strand model there are no two separate things. What remains is: selection by fit (only what returns in phase persists), identity across scale (the same winding at another depth), and constitutive relations (two quantities that are one state, read twice, with no arrow between them). To describe at the right winding depth is to have said everything there is to say.

One more element is needed for this essay: the register.

A single winding, held against its reference, distinguishes exactly three positions. Above. On. Below. Write them $+1, 0, -1$. That is one **trit**. It is not a convention. It is what one winding can carry. Finer distinctions require a second reference, and a second reference is a deeper winding. So the natural number system of the net is balanced ternary: digits $-1, 0, +1$, place value powers of three. In balanced ternary, negation is a mirror (flip every digit), rounding is truncation (drop the deep digits), and zero is exact. No sign bit, no two's complement, no rounding rule. The symmetry is in the digits themselves.

From the register follows the address. An identity in the net is a path of windings: at each depth, one trit. The address of a thing **is** this path, written out. The address at depth n is the first n trits of the address at depth $n+k$. Deeper knowledge refines an address; it never replaces it. Two consequences, both used later: an archive built on computed addresses can grow without ever re-indexing, and two such archives merge by taking the union of their numbers. No authority hands out the addresses. None is needed. The address is calculated from the identity itself.

The depth arithmetic is short. $3^{11} = 177{,}147$: enough distinct addresses for the vocabulary of a large language model. $3^{19} \approx 1.16 \times 10^{9}$: enough for every printed book. $3^{26} \approx 2.54 \times 10^{12}$: every document. $3^{32} \approx 1.85 \times 10^{15}$: every sentence. $3^{42} \approx 1.09 \times 10^{20}$: every moment of every human being who ever lived. Forty-two trits is nine bytes. The address space of everything knowable fits in the width of a machine word.

This is the whole apparatus. Strand, winding, closure, mesh, tension; the trit as register; the address as truncation. Now read the machine.

3. Reading the Machine

The evidence is a single file: `inference/model.py` in the public DeepSeek-V3 repository. It is 808 lines of Python. It is the complete inference core of one of the largest open language models: embedding, twenty-seven transformer layers, attention, a mixture of experts, an output head. Nothing in it is hidden. Every operation named below can be checked against the line that performs it.

The reading finds four net operations. None of them is called by its net name. All four arrived in the code for engineering reasons. That is the point.

3.1 Position is winding

The function `precompute_freqs_cis` encodes the position of a token in a sequence. It does not store position as a coordinate. It stores position as a **rotation**. The code computes frequencies `freqs = 1 / base^(2i/d)` for each pair of embedding dimensions, multiplies them by the position index, and calls `torch.polar` — which places each position on the complex unit circle as a phase angle. A token at position *t* is not "at t". It is wound *t* times, at a different rate for each dimension pair. Fast windings (high frequency) carry local relations. Slow windings carry long-range relations. Same structure, different winding depth: identity across scale, in production code.

The correction functions for long sequences make the winding explicit. `find_correction_dim` takes an argument literally named `num_rotations` and computes at which embedding depth a given number of rotations occurs: `dim · log(max_seq_len / (num_rotations · 2π)) / (2 · log(base))`. The engineers who extended the context window did not reason about coordinates. They reasoned about how many times each dimension winds, and corrected the depths where the winding count changes. This is winding arithmetic, performed daily, under another name.

Absolute position encodings — coordinates on a stage — were the original design, in the 2017 transformer. They lost. Every current major model (DeepSeek, Llama, Qwen, Mistral) uses rotary position: phase, not place.

3.2 The gate is selection by fit

DeepSeek-V3 is a mixture of experts. Each layer holds 64 routed expert modules. For each token, the `Gate` class computes a score against every expert, selects the top 6, and sets every other expert's score to negative infinity through an explicit mask: `scores.masked_fill_(mask, float("-inf"))`. What does not pass gets no passage. Fifty-eight of sixty-four experts contribute exactly nothing to that token. There is no arrow from score to passage. The score and the passage are one movement — a constitutive relation, written in three lines of tensor code.

The balancing of the gate is more telling still. Earlier mixtures forced balance with an auxiliary loss: a second objective pulling against the first. DeepSeek-V3 removed it. Its published balancing method adds a per-expert bias b, adjusted by a counting rule — $b \leftarrow b + u \cdot \text{sign}(1 - f)$, where f is the expert's recent load — used only for selection, never for the output. The learned score is being pushed aside by a deterministic, counting-based rule, because the learned score created tension the system had to discharge. The gate is drifting from score toward fit. It has not arrived. Section 6 derives where the drift ends.

3.3 State lives deep; the surface is a truncation

The attention mechanism (`MLA`, multi-head latent attention) does not cache the full keys and values of past tokens. It compresses them into a latent of rank 512 (`kv_lora_rank = 512`) and stores only that. Every head reads the past by expanding the latent back up. The information lives at a deeper, compressed layer. Reading is shallow unwinding. This is the address rule of Section 2 in tensor form: the representation at depth *n* is a truncation of depth *n+k*. The model does not keep the surface. It keeps the depth and truncates on demand.

The engineering motive was memory: the full cache did not fit. The structural result is identical to the net's rule. Under pressure, the cache became a truncation hierarchy.

3.4 One strand

Every layer of the model performs two operations, and both have the same form: `x = x + attn(norm(x))`, then `x = x + ffn(norm(x))`. There is one continuous residual stream. All twenty-seven layers read from it and write back onto it. The layers are not a chain of causes, each producing the input of the next. They are windings around the same strand. The normalization (`RMSNorm`) bounds the tension of each winding: it divides the state by its own root-mean-square, so no winding can run away with the strand. One strand, bounded tension, twenty-seven winding depths, then a single readout.

This architecture, too, was a selection. Early deep networks passed activations forward as a chain and could not be trained beyond a few layers. The residual stream — one strand that everything writes onto — is what survived.

3.5 What the reading is, and is not

Four operations of the net, then: winding as position, selection as passage, depth with truncation, one strand under bounded tension. None was designed from theory. Each displaced an alternative because the alternative could not hold the tension: coordinates lost to rotations, dense layers lost to selection, full caches lost to truncation, chains lost to the strand. Selection by fit operates on the engineering itself. Whatever does not return in phase with the constraints — energy, memory, gradient flow — disappears from the codebases.

To be precise about the status of this reading: the code was not derived from the net. The correspondence is found by reading, and the same four blocks appear in every current major architecture, so the correspondence is a property of what survives, not of one file. The claim built on it is therefore not "DeepSeek implements the net". The claim is: under sustained selection pressure, the field's machinery has converged on four of the net's operations while still carrying two assumptions the net does not make. Those two assumptions are the subject of the rest of this essay.

4. The Two Remaining Assumptions

Assumption one: the address is assigned. The model's vocabulary is a table of 102,400 tokens, fixed before training by a separate tokenizer. Each token's embedding — its meaning-position in the model — is a free row in a learned matrix of 102,400 × 2,048 = 209,715,200 parameters. The address of a token is whatever row the table happens to hold. It is handed out by an institution (the tokenizer, the training run), it is revisable, and two models with different tables cannot be merged without translation. The library card catalogue, the URL registry and the vector embedding all share this structure: address assigned from outside.

Assumption two: the register is continuous. Every weight and every activation is a floating-point number, sixteen bits wide (bf16) or eight (fp8). Every selection is a softmax over real numbers. The machine holds, at every position, a register deep enough to distinguish tens of thousands of levels — and then spends almost all of that depth on noise. Training maintains this continuous register with gradients; inference pays for it in energy, per token, forever.

The two assumptions are one assumption seen twice. Both place the identity of things outside the net — in a table, in a real number line — instead of computing identity from the strand itself. The next four sections remove them by derivation. Section 7 shows that the removals have already been measured.

5. Thesis One: An Assigned Address Requires an Institution; a Computed Address Does Not

5.1 The derivation

Start from what "assigned" means. An assigned address exists only as an entry in a table. The table must be created, stored, maintained and agreed upon. Whoever holds the table can change it. Therefore an assigned address is revisable, revocable, and valid only within the community that recognizes the table. This is not a defect of implementation. It follows from the definition. Every assigned-address system in history — the Dewey decimal system, the URL, the DOI, the token vocabulary — has required an institution, because the address and the institution are the same fact.

Now the alternative. Let the address be **computed**: a deterministic function from the identity itself to a string of trits, with one property — the address at depth n is the truncation of the address at depth $n+k$. Then, without further assumptions:

1. **No institution.** Anyone who has the thing can compute its address. There is no table to hold, so there is no holder. The address cannot be revoked because there is nothing to revoke it from.

2. **Growth without re-indexing.** New knowledge refines addresses; it never invalidates them. If tomorrow's understanding operates at depth $n+k$, every existing address at depth n remains correct: it is the truncation of the refined one. An archive built this way never migrates. This is a proof, not a policy: truncation is idempotent, so refinement commutes with every address already issued.

3. **Merging is the union of numbers.** Two archives built independently on the same address rule occupy the same address space by construction. Federating them is set union on integers. A collision — the same address with different content — is not a schema conflict. It is a localized disagreement about content, visible at one address, resolvable there.

None of these three properties can be added to an assigned-address system afterwards. They follow from computing the address or they do not exist.

5.2 The numbers

The cost of assignment is measurable in the file read in Section 3. The embedding table: 102,400 rows × 2,048 dimensions = 209.7 million learned parameters. The output head, mapping back to the vocabulary, is of the same order. In smaller models the share is dominant: in the multilingual mT5-Base model, the vocabulary and output matrices amount to 256 million parameters — about 66 percent of the entire model. Two thirds of the machine is the table.

The computed alternative for the same vocabulary: 102,400 identities need $\lceil \log_3(102{,}400) \rceil = 11$ trits, since $3^{11} = 177{,}147$. Learned parameters for the address itself: zero. The meaning still has to be built — but from the address, deterministically, not stored as free rows.

5.3 What has been shown already

The removal of the assigned address is not a proposal. It has been executed, measured and published, in three independent forms.

Byte models. ByT5 (Google, 2022) abolishes the vocabulary. Its "table" is the 256 possible byte values plus three markers: 259 entries. The vocabulary share of the model drops from 66 percent (mT5-Base) to 0.1 percent. Quality on spelling-sensitive and noise-sensitive tasks rises. The cost moves: byte sequences are longer, so computation per text grows. The address became computed; the expense migrated from the table to the strand.

Hashed addresses. CANINE (Google, 2022) drops the tokenizer entirely and maps Unicode characters through fixed hash functions — a computed, deterministic address. Against the table-based mBERT it scores 2.8 F1 points higher on the TyDi QA benchmark with roughly 28 percent fewer parameters, while processing four times as many sequence positions. Hash embeddings (2017) showed the general form: replace a table of K × d free parameters with a small shared codebook addressed by hash functions, reducing embedding parameters by orders of magnitude at equal quality.

Dynamic byte patches. The Byte Latent Transformer (Meta, 2024–2025) is the scale test: 8 billion parameters, 4 trillion training bytes, no vocabulary. Bytes are grouped into patches dynamically, by the entropy of the next byte; average patch length 6 to 8 bytes against 4.4 for the tokenized baseline. Result: quality matching Llama 3 at matched training, with up to half the inference computation — a figure that softens when the patching model's own cost is fully counted, but the parity stands.

The direction of all three results is the same. The table can go. What no one has yet measured is the property that makes computed addresses more than a compression trick: the merge. Two independently trained byte-addressed models share one address space by construction. Their federation should be union, not translation. That experiment — two archives, built apart, joined in one step — has never been run. It is the cheapest open proof in this essay, and it is the one that distinguishes "smaller" from "institution-free". Section 9 returns to it.

6. Thesis Two: Selection by Score Requires a Second Register; Selection by Fit Does Not

6.1 The derivation

A gate that selects by score must hold, beside the addresses of the things it routes, a second store: the scoring weights. In the file of Section 3 this store is a matrix of 64 experts × 2,048 dimensions = 131,072 parameters per layer, times the depth of the model. And the second store creates a second problem. Nothing ties the learned scores to balanced load, so every scored gate needs a third element — an auxiliary loss, or DeepSeek's counting bias — to discharge the tension between what the scores prefer and what the machine can carry. Score, plus correction of score: two registers and a regulator.

Selection by fit needs none of this. Let every routable thing carry a computed address (Thesis One). Define an expert as an address prefix of length m. Passage: the first m trits match. Then:

1. **Zero routing parameters.** The address is already there. The prefix comparison costs m trit comparisons and stores nothing.

2. **Balance by construction.** The 3^m prefixes partition the address space into equal classes. If addresses are well distributed — which a computed address rule guarantees by design — load balance is not regulated. It is a property of the partition.

3. **No discharge machinery.** With no learned score there is no tension between score and load, so there is nothing to regulate. The auxiliary loss and the bias rule are not simplified. They are without object.

6.2 What has been shown already

Deterministic routing is measured. Hash Layers (Meta, 2021) route each token to a fixed expert by hashing the token identity: no routing parameters, no balancing objective, no assignment algorithm. On language modelling benchmarks this parameter-free gate matches the learned gates of Switch Transformers and BASE Layers. The finding, stated plainly: the learned router was carrying less than its cost suggested.

The honest boundary of the result: expert-choice routing (Google, 2022), in which experts select tokens rather than the reverse, beats hashing on downstream quality. Pure balance does not by itself produce specialization. The net's answer is that hashing is the wrong deterministic rule: a hash scatters, a prefix organizes. Prefix routing keeps neighbours together at every depth — the same tokens that share shallow trits share experts, and specialization is the partition itself. That specific rule, prefix passage on computed addresses, is exactly the configuration no published system has tested. It sits at the join of Theses One and Two, and it is the second open proof of Section 9.

Meanwhile the production systems drift. DeepSeek-V3, the largest deployed mixture of experts, already replaced the auxiliary loss with a counting rule and keeps the learned score only for the final ranking. The score's territory shrinks release by release. The derivation above says where the shrinking ends: at the address.

7. Thesis Three: The Register of One Winding Is the Trit

7.1 The derivation

Return to the fishing net. Take one winding of the strand and hold it against its reference. What states can it carry? Above the reference. On it. Below it. Three, exactly. A fourth state would require a second reference — and a second reference is a deeper winding, a new place in the net, not a finer state of this one. So the register of one winding is one trit: $+1, 0, -1$. Registers of greater depth are not deeper numbers at one place. They are more windings.

Now read a continuous weight through this lens. A bf16 number is 16 bits. In trit terms: $16 / \log_2 3 = 16 / 1.585 \approx 10.1$ trits of register depth, pressed into a single storage location and paid for at every multiplication. The derivation's question is empirical: how much of that depth does the machine's function actually use?

7.2 The measurement

The answer, measured at scale by three independent groups, is: about one trit.

BitNet b1.58 (Microsoft, 2024) trains language models whose every weight is ternary: $-1, 0, +1$ — 1.58 bits, which is $\log_2 3$, which is exactly one trit. At 3 billion parameters and 2 trillion training

tokens, the ternary model matches and slightly exceeds its full-precision peer (average benchmark score 74.34 against 73.22 for StableLM-3B), while using 3.55 times less GPU memory and running 2.71 times faster.

BitNet b1.58 2B4T (Microsoft, 2025), the open 2-billion-parameter model trained on 4 trillion tokens, gives the full bill. Non-embedding memory: 0.4 GB, against 2.0 GB for Llama 3.2 1B and 2.6 GB for Qwen2.5 1.5B. CPU latency per token: 29 ms against 48 and 65. Estimated energy per token: 0.028 joule, against 0.258 for Llama and 0.347 for Qwen. On reasoning benchmarks it leads its class on GSM8K (58.38) and WinoGrande (71.90); on the eleven-benchmark average it sits just under the strongest peer (54.19 against Qwen's 55.23) and far above Llama's 44.90. Parity is task-dependent, not universal; the register claim survives that nuance intact.

Spectra / TriLM (Mila and Nolano, 2024–2025) is the independent replication: 54 models trained side by side, float, quantized and ternary, from 99 million to 3.9 billion parameters on 300 billion tokens. The 3.9-billion ternary model matches the 3.9-billion float model on every benchmark while occupying fewer bits than a float model of 830 million parameters. **Falcon-Edge** (TII, 2025) reproduces the recipe again at 1 and 3 billion parameters and releases the training toolkit.

7.3 The constitutive relation

Now place two ratios beside each other. Register depth: 16 bits / 1.58 bits ≈ 10.1. Measured energy per token: 0.258 J / 0.028 J ≈ 9.2. The energy bill and the register depth are the same quantity, read twice. This is not a causal claim — not "fewer bits cause less energy". It is a constitutive relation: the register **is** the energy account. A machine that holds ten trits of depth at every weight pays ten trits of movement at every step, whether the function uses that depth or not. The measurements say the function of a language model uses about one.

Two boundary notes belong here, stated as facts. First, training still runs on continuous shadow weights; the ternary gain is an inference gain today. Second, on graphics processors the gain is partly lost, because the hardware dequantizes ternary weights back to wide registers before multiplying. The full gain appears on machines whose register matches the model's: the dedicated CPU kernels of bitnet.cpp run ternary models up to 6.25 times faster than the float baseline and drive a 100-billion-parameter ternary model at 7.45 tokens per second on a single consumer processor. A matmul-free ternary architecture (UC Santa Cruz, 2024) runs a billion-parameter model on a field-programmable chip at 13 watts, and holds a 13-billion-parameter model in 4.19 GB where the standard transformer needs 48.5. The register argument and the substrate argument meet in Thesis Four.

The history closes a loop. Balanced ternary is not new to machines. The Setun computer, built in Moscow in 1958, ran on exactly this register; fifty were produced. Knuth called balanced ternary "perhaps the prettiest number system of all". It lost to binary for reasons of component supply, not of arithmetic. Sixty-eight years later, the largest models are being retrained onto it — again not from theory, but because the wider register could not hold the tension of the energy bill.

8. Thesis Four: "Hardware" Is the Background Assumption, Renamed

8.1 The derivation

The standing paradigm of computing rests on a split: hardware and software. A dead, neutral carrier on one side; symbols placed upon it on the other. This is the same split physics makes between the fixed vacuum background and the fields that live on it. And it fails for the same reason.

Follow the consequences of the split. If storage and processing are two places, every computation requires transport between them. The transport has a bandwidth, and the bandwidth becomes the wall. The field has already named it: the memory wall — processor speed has grown faster than the channel to memory for four decades, and the gap now dominates the cost of large models. The Spectra authors motivate ternary training from this wall explicitly. The wall is not an engineering accident. It is the tension of the split itself. Two places that are really one, held apart, must pay transport forever.

Now take Theses One to Three together. The address lives in the strand (computed, not tabled). The selection lives in the address (prefix, not score). The register lives in the winding (trit, not float). Then nothing is left for a separate carrier **to do**. There is no table to hold, no score to store, no depth to shuttle. The carrier was never a thing. In net terms: what the paradigm calls hardware is the deep, slow windings of the net, used as a frozen background for the fast ones. The error is treating those deep windings as dead.

So the correction is not "better hardware", and not "other hardware". The correction is: no separate carrier. Computation is what the net does; a computing device is a region of the net whose windings have been arranged to select. The device does not carry the computation. It **is** the computation, at its own depth.

8.2 The field is already leaking out of the split

Three movements, each independently funded and published, each abandoning one piece of the carrier assumption without naming the whole:

In-memory computing. Analog and digital compute-in-memory chips perform the multiplication inside the storage array. Storage and processing collapse into one place; the transport term goes to zero for that operation. The two places admit they were one.

Optical computing. Photonic processors carry the register in the phase of light. Phase is winding — the same quantity Section 3.1 found in the position code, now as physical substrate. Interference performs the selection: what returns in phase adds, what does not cancels. Selection by fit, executed by the physics, at the energy cost of the light itself. This is the substrate line of Right-Brain Computing: an engineering trajectory, not a metaphor.

Thermodynamic and neuromorphic computing. Stochastic and spiking hardware lets relaxation toward equilibrium do the selecting. The matmul-free ternary model of Section 7 has been run on such a platform (Loihi 2) with higher throughput at lower energy than the GPU baseline. The physics is not simulating the computation. It is performing it.

Each movement gives up one wall of the hardware/software split — the place split, the register split, the executor split — and each justifies itself by energy alone. Put beside them the earlier findings: coordinates lost to rotations, scores losing to counts, floats losing to trits, tables losing to bytes. The pattern is one pattern. Under selection pressure, computation is migrating off the assigned background and into the strand. The migration has no theory in the field's own literature. The net supplies it: there was never a background to compute on.

8.3 The chain

One consequence reaches past engineering. The carrier assumption is what makes the long chain necessary: mine, fab, chip, data center, grid, cloud, subscription. Every link exists to build, feed and rent the separate carrier. The measured numbers of Section 7 already shorten the chain at its last links — 0.4 GB and 0.028 joule per token is a model that runs on the machine already on the table, with no connection, no intermediary and no monthly fee. A school without a budget, a village without fiber, an aid post without an IT department: for them the difference is not a benchmark score. It is that the thing works here, now, on what is present, with no one to ask permission. The removal of the carrier assumption, carried through, removes the chain — the same movement this theory makes in its readings of energy, money and institutions.

9. The Objection That Must Be Faced

One measurement in the literature pushes against Thesis Three, and it deserves its own section rather than a footnote.

Ouyang, Ge, Hartvigsen, Zhang, Mi and Wei ("Low-Bit Quantization Favors Undertrained LLMs", ACL 2025) measure quantization-induced degradation across more than 1,500 checkpoints of the Pythia model suite: 160 million to 12 billion parameters, trained up to 300 billion tokens, quantized after training to 2, 3 and 4 bits. They fit a scaling law:

$$\Delta\text{qLoss} = 0.017 \cdot D^{0.525} / (N^{0.226} \cdot P^{5.50})$$

with D the training tokens, N the parameters, P the bit width. The exponent that matters is on D, and it is positive: the damage from a narrow register **grows** with training. In their projection, a 70-billion-parameter model absorbs more than 17 trillion tokens before 4-bit quantization costs a degradation of 0.2; a 405-billion model, nearly 50 trillion. But at the 100-trillion-token budgets the field is approaching, low-bit registers would no longer be free — especially for small models. In a targeted replication at 120 million and 1.2 billion parameters they train BitNet-style ternary models from scratch and observe the same shape: the ternary loss curve tracks the float curve at first, then falls behind, later than post-training quantization but in the same direction.

What the objection establishes, and what it does not, can be stated exactly.

It establishes: the register question has a second axis. Parameter scale was the axis of the original ternary claim (the gap closes as N grows), and on that axis parity is measured to 3.9 billion parameters by two independent groups. Ouyang adds the token axis (the gap opens as D grows), measured to 300 billion tokens for post-training quantization and to 1.2 billion parameters for native ternary training.

It does not establish: that native ternary training fails at scale. The fitted law is a law of **post-training quantization** — carving a narrow register out of a model that spent its whole training in a wide one. Native ternary training is a different object: the model never had the depth to lose. The 4-trillion-token BitNet 2B4T sits thirteen times beyond Pythia's largest measured budget and holds within about one point of its strongest full-precision peer — consistent with a slowly opening gap, and equally consistent with a bounded one. The 100-trillion figures are extrapolation, as the authors state.

So the objection defines the decisive experiment rather than the verdict. Train ternary and float side by side past 4 trillion tokens and watch the difference. If it stays within about one point, the token axis closes and the trit register stands without qualification. If it grows as $D^{0.53}$, the trit register is the register of inference and of bounded training budgets, and deep training keeps a wider register

— which, in net terms, would itself be a finding: training as a process that temporarily needs deeper windings than the function it leaves behind. Either outcome is informative. The essay's theses do not depend on which one arrives; Thesis Three as stated — the **function** of a trained language model uses about one trit of register depth — is what the existing measurements already show.

10. What Has Been Measured, Assembled

The three removals, each measured separately, in one table of record. Every figure appears in the sections above; this is the assembly.

Track one — the address. Assigned-address cost: 210 million parameters of embedding table in the file of Section 3; 66 percent of the whole model in mT5-Base. Computed-address results: ByT5 (vocabulary share to 0.1 percent), CANINE (+2.8 F1 with 28 percent fewer parameters, deterministic hashed addresses), BLT (8 billion parameters, 4 trillion bytes, no vocabulary, parity with Llama 3 at up to half the inference computation), hash embeddings (orders-of-magnitude parameter reduction, 2017).

Track two — the selection. Scored-gate cost: 131 thousand routing parameters per layer plus balancing machinery. Fit results: Hash Layers (zero routing parameters, parity with learned gates); DeepSeek-V3 in production (auxiliary loss replaced by a counting rule); boundary result: expert choice beats hashing, locating the open question at prefix routing, not at determinism itself.

Track three — the register. Trit results: BitNet b1.58 (parity at 3B/2T; 3.55× memory, 2.71× speed), BitNet 2B4T (0.4 GB, 29 ms, 0.028 J per token against 2.0 GB, 48 ms, 0.258 J for its peer), Spectra (independent parity at 3.9B; ternary 3.9B smaller than float 830M), Falcon-Edge (second replication, toolkit released), bitnet.cpp (6.25× on CPU; 100B model at 7.45 tokens/s on one consumer chip), matmul-free (13B model in 4.19 GB against 48.5; FPGA at 13 watts). Historical anchor: Setun, 1958, fifty machines, balanced ternary.

The convergence itself. Rotary position in every major model since 2023. Residual strand universal since 2016. Truncation caches spreading since 2024. Counting rules displacing auxiliary losses since 2024. In-memory, optical and thermodynamic substrates each dissolving one wall of the carrier split. No publication in the field states the four operations as one structure. That statement is this essay's contribution; the measurements are the field's.

11. The Proof Agenda

What is derived, what is measured, what is open — in order of cost, cheapest first.

Open proof one: the merge. Two archives or models, built independently on computed addresses, joined in one step: union of numbers, no schema translation, no registry, collisions surfacing as localized content disagreements. Derived in Section 5; never measured anywhere. Small models suffice. This is the experiment that separates "cheaper" from "institution-free", and it tests the one property no amount of parameter scale can imitate.

Open proof two: prefix passage. A mixture of experts routed by address prefix on computed addresses — the join of Theses One and Two. The published boundary (expert choice beats hashing) predicts exactly where the interesting result lies: whether an organizing deterministic rule

recovers the specialization that a scattering one loses. Zero routing parameters; balance by partition; small scale suffices.

Open proof three: the three-in-one. One model carrying all three removals at once — computed addresses, prefix passage, trit register — against one standard model at equal quality, with the full bill beside it: parameters, gigabytes, joules per token. No published system combines the three. The separate results make the combination plausible; only the combination makes the claim "the two assumptions can go" a measurement instead of a syllogism.

Open proof four: the token axis. Ternary and float trained side by side beyond 4 trillion tokens (Section 9). Capital-intensive; the field will run it, and either outcome sharpens the theory. To be cited when it lands, not funded from here.

Open proof five: the substrate. The energy claim measured on register-matched machines — CPU kernels, FPGA, optical, in-memory — not on GPUs that widen the register back out. Partly done (Section 7.3); to be completed as the substrates mature.

Proofs one and two are within reach of a single builder with small machines. Proof three is a modest training run. The essay's derivations stand on their own; the proofs convert them from derivation to demonstration, in the only currency the theory itself accepts — working passage, not language.

12. Closing

Read once more what happened, in order. Coordinates lost to rotations. Chains lost to one strand. Full caches lost to truncations. Learned balancing lost to counting. Floats are losing to trits. Tables are losing to computed bytes. Storage and processing are collapsing into one place; registers are moving into phase; selection is moving into the physics. Not one of these steps was taken from theory. Each survived because the alternative could not hold the tension of energy, memory and scale.

That is selection by fit, operating on computation itself. The net does not need to be argued into the machines. It is arriving in them, removal by removal, under its own rule: only what returns in phase persists. What the field lacks is the statement of where the movement ends — no assigned address, no second register, no separate carrier — and the two experiments, merge and prefix, that no one runs because no one's theory predicts they matter.

This essay states the endpoint and predicts the experiments. The rest is passage.

Annotated References

Each entry gives the source, why to read it, and where useful, how.

1. **DeepSeek-AI, "DeepSeek-V3 — inference/model.py" (GitHub, deepseek-ai/DeepSeek-V3).** The evidence file of Section 3: 808 lines containing rotary position, the top-k gate, the rank-512 latent cache and the residual strand. *Why read?* Every claim in Section 3 can be checked against a line of this file. *How:* read `precompute_freqs_cis`, `Gate.forward`, `MLA.__init__` and `Block.forward`; the rest is scaffolding.

2. **DeepSeek-AI, "DeepSeek-V3 Technical Report" (arXiv 2412.19437, 2024).** The system around the file: architecture, training, and the production deployment of auxiliary-loss-free balancing. *Why read?* Shows the counting rule running at the largest deployed scale.

3. **Wang, Gao, Zhao, Sun, Dai, "Auxiliary-Loss-Free Load Balancing for Mixture-of-Experts" (arXiv 2408.15664, 2024).** The bias rule b ← b + u·sign(1–f) of Section 3.2. *Why read?* Documents the learned score being displaced by a deterministic count, in the field's own words. Short.

4. **Ma et al., "The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits" (arXiv 2402.17764, 2024).** The original trit-register result: parity at 3 billion parameters, 3.55× memory, 2.71× speed. *Why read?* The parameter-axis scaling claim starts here. Read together with entry 5 for the measured bill.

5. **Ma et al., "BitNet b1.58 2B4T Technical Report" (arXiv 2504.12285, 2025).** The open ternary model at 2 billion parameters and 4 trillion tokens. *Why read?* Tables 1–2 hold the figures used throughout: 0.4 GB, 29 ms, 0.028 J per token, and the benchmark averages with their honest nuance.

6. **Kaushal, Vaidhya et al., "Spectra: Surprising Effectiveness of Pretraining Ternary Language Models at Scale" (ICLR 2025, arXiv 2407.12327); Spectra 1.1 (arXiv 2506.23025).** The independent replication: 54 models, float against ternary, to 3.9 billion parameters. *Why read?* Strongest evidence for the parameter axis outside Microsoft, and the memory-wall motivation stated by working engineers.

7. **Falcon-Edge team (TII), "Falcon-Edge: 1.58-bit language models" (Hugging Face blog and models, 2025).** Second independent reproduction, with the training toolkit released. *Why read?* Establishes that the ternary recipe travels between labs. Brief.

8. **Ouyang, Ge, Hartvigsen, Zhang, Mi, Wei, "Low-Bit Quantization Favors Undertrained LLMs" (ACL 2025, arXiv 2411.17691).** The objection of Section 9, with the fitted law ΔqLoss = 0.017·D^0.525/(N^0.226·P^5.50). *Why read?* Defines the token axis and the decisive experiment. Read §4.4 for the native-ternary replication; keep the PTQ/native distinction in view.

9. **Zhu, Han, Mao, Dally, "Trained Ternary Quantization" (ICLR 2017, arXiv 1612.01064).** The 2017 ancestor: ternary weights at ~16× compression with equal or better accuracy on image tasks. *Why read?* Shows the trit register predates the language-model era by seven years. Short.

10. **Zhu et al., "Scalable MatMul-free Language Modeling" (NeurIPS 2024, arXiv 2406.02528).** Ternary weights plus element-wise operations; 13B in 4.19 GB against 48.5; FPGA at 13 watts; a neuromorphic run. *Why read?* The closest existing system to a multi-removal machine, and the bridge to Thesis Four's substrates.

11. **Wang et al., "bitnet.cpp: Efficient Edge Inference for Ternary LLMs" (ACL 2025, arXiv 2502.11880).** The register-matched kernels: up to 6.25× over float on CPU; 100B ternary at 7.45 tokens/s on one consumer chip. *Why read?* Converts the register argument into a machine on a desk. Relevant to the chain argument of Section 8.3.

12. **Xue et al., "ByT5: Towards a Token-Free Future with Pre-trained Byte-to-Byte Models" (TACL 2022, arXiv 2105.13626).** The vocabulary abolished: 66 percent of mT5-

Base shown to be table, then removed. *Why read?* The cleanest measurement of what an assigned address costs. Read Table 1 first.

13. **Clark et al., "CANINE: Pre-training an Efficient Tokenization-Free Encoder" (TACL 2022, arXiv 2103.06874).** Deterministic hashed character addresses beating the table (+2.8 F1, 28 percent fewer parameters). *Why read?* The computed address demonstrated, not merely proposed. Short; complementary to ByT5.

14. **Pagnoni et al., "Byte Latent Transformer: Patches Scale Better Than Tokens" (ACL 2025, arXiv 2412.09871).** The scale test of the computed address: 8B, 4T bytes, parity with Llama 3. *Why read?* Shows the vocabulary staying gone at scale; note the independent re-analysis that tempers the 50-percent inference figure.

15. **Svenstrup, Hansen, Winther, "Hash Embeddings for Efficient Word Representations" (NeurIPS 2017, arXiv 1709.03933).** The general mathematics of replacing K×d tables with hashed codebooks. *Why read?* The parameter arithmetic behind Thesis One, eight years early.

16. **Roller, Sukhbaatar, Szlam, Weston, "Hash Layers for Large Sparse Models" (NeurIPS 2021).** Routing with zero parameters, matching learned gates. *Why read?* The measured half of Thesis Two. Keep beside entry 17.

17. **Zhou et al., "Mixture-of-Experts with Expert Choice Routing" (NeurIPS 2022).** The boundary: expert choice beats hashing. *Why read?* Locates the open experiment precisely — the failure of scattering, not of determinism. Honest reading of both papers together is the point.

18. **Su et al., "RoFormer: Enhanced Transformer with Rotary Position Embedding" (arXiv 2104.09864, 2021).** Where position became rotation. *Why read?* The origin of the winding in Section 3.1, adopted since by every major model.

19. **Brusentsov and the Setun literature; Knuth, "The Art of Computer Programming", Vol. 2, on balanced ternary.** The 1958 balanced-ternary computer, fifty machines built, and the arithmetic's properties. *Why read?* The trit register has run before; its loss was industrial, not mathematical. Knuth's treatment is three pages and complete.

20. **Wang, Ma, Wei, "BitNet a4.8: 4-bit Activations for 1-bit LLMs" (arXiv 2411.04965, 2024).** Activations following weights down: 55 percent active parameters, 3-bit cache. *Why read?* Shows the register migration continuing past the weights, toward the whole machine. Brief.