Worker-Verifier Model Identity Verification with Logprobs and Routed Experts: An Experimental Study

Workers generate answers; Verifiers use teacher-forced prefill on the same inputs and outputs to compare logprobs and expert routes across hardware, model sizes and quantization configurations.

Main findings

In these tests, Dense logprob deviations separate same-model replay from quantization and model substitution more clearly. MoE logprobs have heavy-tailed errors; layer-0 route fingerprints offer better separation than all-layer aggregation in the reported route experiments.

These are statistical model-identity experiments, not cryptographic proofs of complete inference. Results apply to the reported test configurations.

Abstract

This paper studies whether token-level logprob traces recorded during worker generation can be used for effective model identity verification under a worker-verifier architecture. The experiments cover two model families: dense Transformer models and Mixture-of-Experts (MoE) models. For dense models, we use `Qwen3-32B BF16` and `Qwen3-8B BF16` as the main targets, and compare same-model BF16 verifiers, FP8/AWQ quantized verifiers, and smaller-model verifiers using selected/top-k logprob differences. The results show that dense models have a narrow and stable same-model logprob baseline, while FP8, AWQ, and smaller models separate clearly from that baseline. Therefore, logprob fingerprints can effectively distinguish dense model identities.

For MoE models, we use `Qwen3.6-35B-A3B` and compare BF16 workers against BF16/FP8 verifiers, as well as cross-verification between BF16 and FP8 workers. The experiments show that MoE logprob differences do contain a statistical right shift from BF16 to FP8, but the same-model BF16 baseline itself has sparse, position-sensitive extreme spikes. As a result, token-level logprob alone cannot form a clean and robust hard decision boundary. Further routed experts experiments show that expert routing traces provide a more direct structural fingerprint for MoE models: same-model and cross-model cases differ consistently in entry mismatch, token-layer set match, and early-layer route match metrics, and these differences remain stable across 6000ws/H100 cross-GPU comparisons. A particularly important finding is that the layer-0 route fingerprint is clearer than aggregating all layers, because it has not yet been diluted by later-layer routing error accumulation and path branching. Therefore, dense models can rely primarily on logprob verification, while MoE models should use routed_experts, especially the layer-0 route fingerprint, as the main verification path.

Keywords: worker-verifier, logprob fingerprint, MoE, routed experts, model identity verification, vLLM

1. Introduction

In open inference networks or decentralized compute networks, verifying whether a worker actually ran the claimed model is a core requirement for trustworthy model services. Recomputing full outputs is expensive, while checking only the final text is insufficient to distinguish models from the same family, quantized variants, or smaller models pretending to be larger ones. A natural approach is to require the worker to submit a logprob trace along the generated path, and then let the verifier perform teacher-forced replay on the same `input_ids + output_ids` path. The verifier then compares per-token logprobs, ranks, and top-k distributions.

This paper focuses on two questions:

1. For dense Transformer models, is the logprob trace sufficient as a model identity fingerprint?
2. For MoE models, is the logprob trace equally effective? If not, can expert routing information provide a better verifier?

Our conclusions are architecture-dependent:

- Dense models: logprob fingerprints are effective. The same-model BF16 baseline is stable, while FP8, AWQ, and smaller models deviate clearly.
- MoE models: logprob fingerprints contain signal, but are not clean enough. The same-model baseline has heavy-tailed spikes, making single-token or simple percentile thresholds prone to error.
- MoE routed_experts: expert routing traces provide a structural fingerprint that can distinguish model or quantization variants more directly than logprobs.

2. Worker-Verifier Method

2.1 Worker Evidence

The worker performs normal decode generation and records the following evidence:

```
input_ids
output_ids
worker_selected_logprob[d]
worker_selected_rank[d]
worker_topk_logprobs[d]
```

Here, `d` is the output token depth. The worker generation stage can use greedy decoding or sampling. In the main experiments, we use:

```
temperature = 0.0
top_p = 1.0
top_k = 0
max_new_tokens = 128
```

2.2 Verifier Replay

The verifier does not regenerate text. Instead, it fixes the path submitted by the worker:

```
full_prompt_ids = input_ids + output_ids
```

It then uses vLLM prompt logprobs to perform full-prefill replay. For output token `output_ids[d]`, the verifier alignment position is:

```
prompt_pos = len(input_ids) + d
```

Therefore, the comparison is between the model distributions produced by the worker and verifier on the exact same token path, rather than a text similarity comparison between two freely generated outputs.

2.3 Logprob Metrics

This paper mainly uses the following metrics:

Metric	Meaning
`abs_logprob_diff`	`abs(worker_selected_logprob - verifier_selected_logprob)`
`p95/p99/p999`	Percentiles of `abs_logprob_diff`
`rank_delta_rate`	Fraction of selected tokens whose rank changes
`topk_jaccard`	Jaccard overlap between worker/verifier top-k token sets
`union_js`	Jensen-Shannon divergence over the top-k union
`missing_selected_count`	Number of worker selected tokens not returned by the verifier logprobs

It is important to emphasize that these experiments compare selected/top-k logprobs exposed by vLLM, not full-vocabulary raw logits.

3. Dense Model Experiments

For dense models, the GPU comparison is mainly in the logprob verifier dimension: Qwen3-32B covers 6000ws and H100 NVL verifiers, while Qwen3-8B covers 4090 and L4 verifiers. Therefore, the dense-model conclusion compares not only model and quantization differences, but also whether the same-model logprob baseline remains stable across GPUs.

3.1 Qwen3-32B Setup

The Qwen3-32B dense experiment uses 300 prompts covering the following input token buckets:

```
32, 128, 512, 2048, 8192, 16384
```

Each bucket has 50 prompts. The worker is:

```
Qwen/Qwen3-32B BF16
GPU: 6000ws
backend: vLLM 0.23.0
batch size: 8
logprobs: 64
output rows: 36882
```

The verifiers cover:

- `Qwen3-32B BF16`
- `Qwen3-32B-FP8`
- `Qwen3-32B-AWQ`
- `Qwen3-14B BF16`
- `Qwen3-8B BF16`
- verifier GPUs: 6000ws and H100 NVL

3.2 Qwen3-32B Results

verifier GPU	verifier model	quant	mean	p95	p99	p999	rank_delta_rate	jaccard p05	union_js p99
6000ws	Qwen3-32B	BF16	0.0102	0.0598	0.1167	0.2478	0.0081	0.8824	0.0035
H100 NVL	Qwen3-32B	BF16	0.0099	0.0583	0.1140	0.2243	0.0070	0.8824	0.0033
6000ws	Qwen3-32B-FP8	FP8	0.0268	0.1370	0.2997	0.7665	0.0205	0.7067	0.0217
H100 NVL	Qwen3-32B-FP8	FP8	0.0354	0.1847	0.4021	1.0112	0.0271	0.6410	0.0336
6000ws	Qwen3-32B-AWQ	AWQ	0.0671	0.3365	0.7962	2.2053	0.0478	0.5238	0.0921
6000ws	Qwen3-14B	BF16	0.2608	1.3907	3.9261	8.1296	0.1165	0.3061	0.4250
6000ws	Qwen3-8B	BF16	0.3749	2.0618	5.8105	11.6767	0.1402	0.2549	0.5189

Using `6000ws worker -> 6000ws Qwen3-32B BF16 verifier` as the baseline:

challenger	p99 ratio	p999 ratio	rank_delta ratio
Qwen3-32B-FP8	2.57x	3.09x	2.54x
Qwen3-32B-AWQ	6.82x	8.90x	5.92x
Qwen3-14B	33.63x	32.81x	14.42x
Qwen3-8B	49.77x	47.12x	17.35x

The results show that same-model BF16 forms a narrow valid drift envelope on both 6000ws and H100 NVL. FP8 already deviates significantly, while AWQ and smaller models deviate even more strongly. Therefore, for dense models such as Qwen3-32B, selected/top-k logprob traces form an effective model identity fingerprint.

3.3 Qwen3-8B Results

The Qwen3-8B experiment uses 200 prompts covering the following input buckets:

```
32, 128, 512, 2048
```

The worker is `Qwen3-8B BF16` on a 4090 GPU. The verifiers cover 4090 and L4, and include Qwen3-8B BF16, Qwen3-8B-FP8, Qwen3-8B-AWQ, and Qwen2/Qwen2.5-7B models.

verifier GPU	verifier model	quant	runs	mean p99	mean abs diff	mean rank_delta	mean union_js_p99
4090	Qwen3-8B	BF16	4	0.1018	0.0081	0.0060	0.0028
L4	Qwen3-8B	BF16	3	0.1033	0.0090	0.0071	0.0033
4090	Qwen3-8B-FP8	FP8	3	0.3123	0.0290	0.0191	0.0229
L4	Qwen3-8B-FP8	FP8	3	0.3065	0.0273	0.0192	0.0225
4090	Qwen3-8B-AWQ	AWQ	2	1.1339	0.0800	0.0494	0.1436
L4	Qwen3-8B-AWQ	AWQ	3	1.1358	0.0799	0.0495	0.1431
4090	Qwen2.5-7B	BF16	2	5.8311	0.5703	0.1954	0.6279
4090	Qwen2-7B	BF16	2	6.3173	0.6087	0.2036	0.6281

The Qwen3-8B results further confirm that dense models have a narrow same-model BF16 baseline, small cross-GPU differences, and clear deviations for FP8/AWQ/other models. Based on these results, reasonably clear batch-level pass/reject thresholds can be defined, for example:

```
BF16 pass:
mean_abs_logprob_diff <= 0.015
p95_abs_logprob_diff <= 0.090
p99_abs_logprob_diff <= 0.200
rank_delta_rate <= 0.012
topk_jaccard_mean >= 0.940
union_js_p99 <= 0.012

Reject:
mean_abs_logprob_diff > 0.020
p95_abs_logprob_diff > 0.120
p99_abs_logprob_diff > 0.250
rank_delta_rate > 0.030
topk_jaccard_mean < 0.920
union_js_p99 > 0.020
```

4. MoE Logprob Experiments

4.1 Qwen3.6-35B-A3B Setup

The MoE experiments use `Qwen/Qwen3.6-35B-A3B` and include two main datasets:

1. A 300-sample experiment for detailed analysis of BF16 baseline spikes, FP8 right shift, and window-level decision metrics.
2. A 5000-sample `moe_1` experiment to confirm BF16/FP8 worker-verifier directionality at larger scale.

4.2 300-Sample MoE Results

run	verifier quant	mean	p50	p95	p99	p999	max	rank_delta_rate	jaccard p05	union_js p99
BF16 original	BF16	0.0591	0.000606	0.1432	0.4331	11.1895	24.1029	0.0251	0.7297	0.0582
BF16 replicate 0001	BF16	0.0540	0.000525	0.1196	0.3665	10.1542	25.4778	0.0224	0.7297	0.0530
BF16 replicate 0002	BF16	0.0572	0.000568	0.1362	0.4208	10.5622	25.5559	0.0237	0.7297	0.0557
BF16 replicate 0003	BF16	0.0622	0.000619	0.1423	0.4181	11.8944	24.1498	0.0246	0.7297	0.0636
FP8 verifier	FP8	0.0848	0.001088	0.2278	0.6729	12.6798	27.5559	0.0364	0.6842	0.1165

From mean, p95, p99, rank_delta_rate, and union_js_p99, the FP8 verifier does show a statistical right shift relative to the BF16 verifier. However, the MoE same-model BF16 baseline already has an extreme heavy tail:

```
BF16 p50 approx. 5e-4
BF16 p99 approx. 0.36 - 0.43
BF16 p999 approx. 10 - 12
BF16 max approx. 24 - 26
```

In other words, most tokens have very small differences, but a small number of tokens can produce extremely large logprob differences. This makes it difficult for a MoE logprob verifier to form a clean hard decision boundary using single-token thresholds or simple percentiles.

4.3 MoE Spike Pattern

The four BF16 baselines show:

threshold	BF16 token rate	pattern
`abs_diff > 1`	0.53% - 0.62%	92.8% - 94.3% are isolated single-point runs
pooled BF16 `p99=0.4082`	0.89% - 1.07%	about 90% of spike runs are single-point
pooled BF16 `p999=11.123`	0.086% - 0.115%	96.9% - 100% of spike runs are single-point

These spikes are not purely random noise. Using `(sample_id, depth)` as the location, among spike locations with `abs_diff > 1`:

condition	ratio
Appears in at least 2 BF16 runs	65.1%
Appears in at least 3 BF16 runs	43.6%
Appears in all 4 BF16 runs	21.5%

This suggests that spikes are tied to specific tokens or contexts, likely related to MoE routing boundaries, decode/prefill computation path differences, or kernel scheduling. Regardless of the root cause, they destroy the clean decision boundary required by a logprob verifier.

4.4 Window-Level Statistics Help, But Do Not Fully Solve the Problem

Using window medians can reduce the BF16 tail:

run	token p99	window=5 median p99	window=16 median p99
BF16 original	0.4331	0.1101	0.0513
BF16 0001	0.3665	0.0972	0.0437
BF16 0002	0.4208	0.1131	0.0540
BF16 0003	0.4181	0.1124	0.0537
FP8	0.6729	0.1685	0.0802

When using the pooled BF16 window p99 as the threshold:

window	BF16 bad-rate range	FP8 bad rate
3	0.79% - 1.08%	2.99%
5	0.78% - 1.13%	3.41%
16	0.65% - 1.17%	4.66%
32	0.50% - 1.22%	5.07%

This shows that window-level statistics can reveal the systematic FP8 shift. However, this remains a statistical separation rather than a clean and stable model identity boundary like the one observed for dense models. MoE logprobs provide useful evidence, but should not be used as the sole final identity verifier.

4.5 5000-Sample MoE Results

The large-scale `moe_1` experiment contains 5000 prompts with the following input buckets:

```
32, 128, 512, 2048, 8192
```

Each bucket has 1000 prompts. Both worker and verifier run on 6000ws, with vLLM version 0.27.1 and logprobs/top-k set to 64.

worker evidence	verifier	runs	rows	mean	p50	p95	p99	p999	max	rank_rate	union_js_p99
BF16 worker	BF16 verifier	3	531030	0.0816	0.000656	0.1700	0.6349	14.6015	32.1399	0.0305	0.1252
BF16 worker	FP8 verifier	1	531030	0.0873	0.000952	0.2298	0.7324	13.5771	29.3174	0.0371	0.1273
FP8 worker	BF16 verifier	3	536996	0.0974	0.000725	0.2362	0.8344	15.4485	28.2228	0.0386	0.1565

These results reinforce the earlier conclusion: MoE logprobs can reveal an overall distributional difference between BF16 and FP8, but the same-model BF16 baseline already has large p99/p999/max values. As a result, FP8 and BF16 do not separate as cleanly as they do for

dense models.

5. Routed Experts Experiments

MoE also has experiments comparing different GPUs, but the core evidence comes from routed_experts rather than the 5000-sample logprob overall results above. The route reports cover BF16/FP8 same-model and cross-model comparisons across 6000ws and H100, and are used to verify whether expert routing fingerprints remain distinguishable across GPUs.

5.1 Method

In addition to the output token distribution, a MoE model selects routed experts for each token at each MoE layer. When vLLM is enabled with:

```
--enable-return-routed-experts
```

it can return the expert routing result for each token/layer. The experiment compares:

```
decode-time routed_experts
vs
prefill-time routed_experts
```

and computes:

Metric	Meaning
`entry_mismatch_rate`	Order-sensitive mismatch for token/layer/top-k slots
`token_layer_set_match_rate`	Whether the top-k expert set matches for each token/layer
`token_match_rate`	Whether all layer/top-k routes of a token match
`mean_topk_jaccard`	Average Jaccard similarity of routed expert top-k sets

5.2 Routed Experts Results

case	same_model	cross_model	samples	layers	tokens	entry_mismatch_rate	token_layer_set_match_rate	mean_topk_jaccard
6000_bf16_vs_bf16	1000	0	1000	all	123616	0.1919	0.7690	0.9430
6000_bf16_vs_h100_bf16	1000	0	1000	all	123616	0.1910	0.7700	0.9433
h100_bf16_vs_bf16	1000	0	1000	all	123921	0.1889	0.7722	0.9438
6000_fp8_vs_fp8	1000	0	1000	all	123695	0.2725	0.6701	0.9164
6000_fp8_vs_h100_fp8	1000	0	1000	all	123695	0.2730	0.6692	0.9163
h100_fp8_vs_fp8	1000	0	1000	all	124354	0.2658	0.6784	0.9188
6000_bf16_vs_fp8	0	1000	1000	all	123616	0.2922	0.6452	0.9097
6000_bf16_vs_h100_fp8	0	1000	1000	all	123616	0.2915	0.6462	0.9100
h100_bf16_vs_fp8	0	1000	1000	all	123921	0.2941	0.6426	0.9088
6000_fp8_vs_bf16	0	1000	1000	all	123695	0.2926	0.6446	0.9092
6000_fp8_vs_h100_bf16	0	1000	1000	all	123695	0.2928	0.6443	0.9091
h100_fp8_vs_bf16	0	1000	1000	all	124354	0.2932	0.6434	0.9088

The all-layer route results show that MoE routed_experts already covers same-model and cross-model comparisons across 6000ws/H100:

- BF16 same-model: entry mismatch is about `0.189-0.192`
- FP8 same-model: entry mismatch is about `0.266-0.273`
- BF16/FP8 cross-model: entry mismatch is about `0.292-0.294`

Therefore, routed experts can distinguish BF16 same-model from BF16/FP8 cross-model cases. FP8 same-model and cross-model are closer under all-layer metrics, but can be further separated using early-layer metrics.

5.3 Early-Layer Route Fingerprint

When observing only the first few layers, same-model and cross-model cases separate more clearly:

case	layers	entry_mismatch_rate	token_layer_set_match_rate	token_match_rate	mean_topk_jaccard
6000_bf16_vs_fp8_layers_0	0	0.1134	0.8871	0.5912	0.9744
6000_bf16_vs_fp8_layers_0_1	0,1	0.1318	0.8621	0.3086	0.9685
6000_bf16_vs_fp8_layers_0_4	0-4	0.1694	0.8111	0.0292	0.9563
6000_fp8_vs_fp8_layers_0	0	0.0492	0.9504	0.8009	0.9890
6000_fp8_vs_fp8_layers_0_1	0,1	0.0764	0.9190	0.5085	0.9820
6000_fp8_vs_fp8_layers_0_4	0-4	0.1262	0.8592	0.0756	0.9680

At layer 0, FP8 same-model has an entry mismatch of `0.0492`, while BF16/FP8 cross-model has `0.1134`; token-layer set match also drops from `0.9504` to `0.8871`. This shows that the early-layer routed_experts route can serve as a stronger structural fingerprint.

More importantly, layer 0 is not merely a reduced-layer approximation that happens to work; in the current data, it is better suited for MoE identity verification than all-layer aggregation. In all-layer metrics, FP8 same-model entry mismatch is about `0.266-0.273`, while BF16/FP8 cross-model is about `0.292-0.294`, a gap of only about `0.02`. At layer 0, however, FP8 same-model is `0.0492` and BF16/FP8 cross-model is `0.1134`, increasing the absolute gap to `0.0642`, with a cross/same ratio of about `2.30x`. The BF16 verifier layer-0 cross-GPU check shows the same trend: BF16 positives have single-sample means around `0.024-0.029`, while FP8 negatives are around `0.115-0.117`, producing a much clearer separation.

The cross-GPU random-5 threshold test further supports this conclusion. Using only layer 0 and aggregating entry mismatch over 5 randomly sampled prompts each time, the BF16 verifier with threshold `0.060` achieves a combined TPR of `1.000000` and a negative false pass rate of `0.000000`. The FP8 verifier with threshold `0.080` achieves a TPR of `0.999453`, while the negative false pass rate remains `0.000000`. This shows that the layer-0 route fingerprint is not an artifact of a single report, but remains stable under 6000ws/H100 cross-GPU comparisons.

5.4 Why Layer 0 Outperforms All Layers

At first glance, using all MoE layers might appear to contain more information. However, the experiments show that all-layer aggregation mixes identity signal with path noise. There are three main reasons.

First, the routing input to later MoE layers has already been affected by earlier layer outputs. Even for the same model, decode-time versus prefill-time execution, small hardware/kernel numerical differences, batch composition, and similar factors can move some tokens across a router decision boundary in later layers. These later-layer mismatches raise the same-model baseline and increase the all-layer entry mismatch.

Second, averaging over all layers dilutes the most discriminative early-layer differences. Layer 0 acts directly on the embedding and original context representation, before multiple rounds of expert selection have propagated errors. It is therefore closer to an initial routing signature: same-model layer-0 routes are more stable, while BF16/FP8 or different-model router boundary differences are already visible.

Third, cross-layer token-level exact match is an overly strict metric. As the number of layers grows, a boundary flip in any single layer can reduce token_match_rate, quickly pushing all-layer token_match into a low-value region and reducing discriminative power. In contrast, layer-0 entry mismatch and token-layer set match preserve a larger dynamic range, making thresholds easier to set.

Therefore, for MoE routed_experts verifiers, we recommend using layer 0 as the primary criterion or a strong feature. All-layer route diff is better suited as an auxiliary diagnostic for understanding the source of routing drift, rather than as a replacement for the layer-0 identity fingerprint.

5.5 Why Routed Experts Are Better Suited for MoE

MoE output logprobs are affected by expert routing, batch composition, and decode/prefill path differences. Even when the model weights are identical, if some tokens lie near a routing boundary, the selected logprob can produce extreme spikes. Logprob is the final projection of the output distribution and cannot directly explain whether these spikes come from model differences, path differences, or routing-sensitive points.

routed_experts directly exposes a structural intermediate state of the MoE model:

```
input/context -> router -> selected experts
```

If two models, two quantization settings, or two execution paths differ in expert selection, this difference appears before the final logprob. Thus, routed_experts is a more natural verifier for MoE identity: it verifies the key internal decision of the MoE architecture, rather than only the resulting output probabilities.

6. Discussion

6.1 The Fundamental Difference Between Dense and MoE Models

In dense models, every token passes through the same dense layers. Numerical differences mainly come from GPU, kernel, dtype, or quantization effects. Therefore, the same-model BF16 logprob replay baseline is usually narrow, while smaller models, quantization changes, or weight changes cause stable distribution shifts over many tokens.

In MoE models, each token also goes through a router that selects experts. Even when the final output token path is the same, decode-time versus prefill-time routing, batch composition, or numerical path differences can move a small number of tokens into different expert sets, producing local logprob spikes. This makes the MoE logprob baseline naturally wider and more heavy-tailed.

6.2 Logprobs Are Still Useful, But Not the Main MoE Criterion

The MoE logprob experiment is not useless. FP8 shows statistical shifts relative to BF16 in mean, p95, p99, rank_delta_rate, union_js_p99, and bad-window-rate. It can serve as an auxiliary signal, especially for window-level and batch-level risk assessment.

However, if the goal is clean identity verification in production, MoE logprobs have several issues:

1. The same-model baseline itself has an extreme long tail.
2. Large-diff locations recur, indicating that they are not simple random noise.
3. FP8/BF16 differences are partially masked by the same-model spike envelope.
4. Single-token and simple percentile thresholds can easily oscillate between false reject and false accept.

Therefore, MoE should not rely only on a logprob verifier.

6.3 Recommended Verification Strategy

For dense models:

```
Use selected/top-k logprob replay.
Build a multi-metric envelope using mean/p95/p99/rank_delta/topk_jaccard/union_js.
```

For MoE models:

```
Primary criterion: layer-0 routed_experts route fingerprint.
Auxiliary diagnostic: all-layer route drift.
Secondary signal: window-level logprob statistics.
```

A two-stage workflow can be used:

1. The worker submits `input_ids + output_ids + logprob trace + routed_experts trace`.
2. The verifier performs prefill replay on the same path and compares:
3. layer-0 routed_experts mismatch / set match / random-5 aggregate score;
4. all-layer routed_experts route set / Jaccard / mismatch rate;
5. window-level drift of selected/top-k logprobs;
6. missing selected tokens and rank drift.

7. Limitations

1. The logprob experiments use selected/top-k logprobs exposed by vLLM, not full-vocabulary raw logits.
2. The output length limit is mainly 128 tokens; long-output settings require separate calibration.
3. The main Dense 32B experiment uses `6000ws worker evidence` and does not cover every worker/verifier direction.
4. MoE logprobs already fail to form a clean and robust decision boundary in the same-GPU setting, so cross-GPU logprob verification is not a key missing experiment required for the paper's conclusion. MoE cross-GPU stability should mainly be verified through routed_experts, especially the layer-0 route fingerprint.
5. routed_experts experiments already cover 6000ws/H100 cross-GPU comparisons. For production deployment, the vLLM version, prefix cache, batching policy, and routed_experts return format should still be fixed or calibrated.
6. This paper studies statistical identity verification, not a cryptographic proof.

8. Conclusion

The core conclusions of this paper are:

```
Dense models can be effectively distinguished using worker-verifier logprob fingerprints;
MoE models cannot be cleanly and robustly distinguished using logprobs alone;
For MoE models, layer-0 routed_experts provides a more effective structural model fingerprint.
```

Experimentally, the Qwen3-32B and Qwen3-8B dense results show that the same-model BF16 baseline is narrow and stable across GPUs, while FP8, AWQ, and smaller models separate clearly from the baseline. In contrast, the Qwen3.6-35B-A3B MoE results show that the same-model BF16 baseline already has heavy-tailed spikes. FP8 has a statistical right shift, but the separation is not as clean as in dense models. Further routed_experts experiments show observable structural differences between same-model and cross-model routing traces, and this conclusion remains valid under 6000ws/H100 cross-GPU route comparisons. More importantly, layer 0 has a higher signal-to-noise ratio than all-layer aggregation: it preserves early routing differences while avoiding later-layer path branching and accumulated mismatches that raise the same-model baseline.

Therefore, a worker-verifier system should be designed by model architecture: dense models should primarily use logprob fingerprints, while MoE models should primarily use layer-0 routed_experts fingerprints, with all-layer route drift and logprob window statistics as auxiliary signals.

Appendix: Main Data Sources

Data	Path
Dense 32B worker-verify	`worker_logits_verify/results/qwen3_32b_identity_v1/worker_verify/`
Qwen3-8B worker-verify	`qwen3_8b_4090_data_results/results/qwen3_8b_identity_v1/worker_verify/`
MoE 300-sample worker-verify	`results/qwen3.6-35b-a3b-bf16-6000ws/worker_verify/`
MoE BF16 replicates	`replicates/results/qwen3.6-35b-a3b-bf16-6000ws/worker_verify/`
MoE 5000-sample data	`moe/moe_1_data.zip`
MoE 5000-sample results	`moe/moe_1_results.zip`
Routed experts reports	`moe/reports/`
Full experiment data document	`worker_verifier_experiment_data_document.md`

The appendix preserves data-path identifiers from the original report; they are not download links on this website.

Back to papers and experiments →