

Worker vs. Verifier Latency Benchmark Report

Single-request latency for worker generation against verifier prefill on Qwen3-8B and one RTX 4090: 48 input x output combinations, per-token decode cost, prefill throughput and the worker/verifier cost ratio.

CITATION

These experiments, figures and data files are published by the **TrueOpen.ai team**. When citing or redistributing them, in whole or in part, state that the source is the TrueOpen.ai team and link to www.trueopen.ai. The same requirement is repeated in the download bundle's README, in CITATION.txt, and in the header of every CSV file.

This report is based on the timing results from `vllm_worker_verify_pipeline/benchmark_worker_verifier_latency.py`. It quantifies the time-cost gap between the two paths and analyzes how input / output length affects that gap.

- Script: `vllm_worker_verify_pipeline/benchmark_worker_verifier_latency.py`
- Model: Qwen/Qwen3-8B (quant=bf16, dtype=bfloat16)
- Hardware: single RTX 4090, tensor_parallel_size=1
- Sampling: top_k logprobs = 20, warmup=2, repeats=5, batch=1
- Backend: vLLM offline LLM class (in-process, V1 EngineCore)

Both paths share a single `vllm.LLM` instance (same model, same GPU state):

- **worker**: normal generation, `prefill(input) + decode(output_len), SamplingParams(max_tokens=output_len, logprobs=20)`; output length is pinned exactly via `ignore_eos + min_tokens`.
- **verifier**: a single prefill over the worker's `input_ids + output_ids` to recover per-position top-k logprobs, `SamplingParams(max_tokens=1, prompt_logprobs=20)`.

1. Executive Summary

Core conclusion: the verifier's cost is far below the worker's generation cost — in typical multi-token output scenarios, verifying a full generation takes only about 1% of the cost to produce it (worker/verifier $\approx 80\times$ – $114\times$). This is direct evidence that the worker/verifier protocol is viable in terms of time cost.

The gap stems from the fundamentally different nature of the two paths. The worker spends nearly all its time on **serial decode**, emitting tokens one at a time at roughly 17.5 ms per token, essentially independent of input length. The verifier performs a **single parallel prefill**, computing all positions of input+output at once at roughly 5,200 tok/s. The ~two-orders-of-magnitude efficiency gap between serial decode and parallel prefill is the source of the cost difference. Consequently, the longer the output, the worse the worker fares and the more favorable verification becomes: the ratio peaks at 80×–114× when output is 1024–2048.

The only exception is when the output is extremely short (=1), where the verifier is actually slightly slower (ratio 0.62–0.82) — it must compute top-k logprobs at every position, and that fixed overhead slightly outweighs the tiny decode savings. This is a corner case and does not affect the main conclusion.

2. Worker Cost Model: Output-Dominated, Serial Decode

With input fixed at 128, worker time scales almost perfectly linearly with output, and the per-token cost holds steady at ~17.5 ms:

output	worker med (s)	per token (ms)
1024	17.63	17.2
2048	35.40	17.3
4096	71.49	17.5
8192	145.77	17.8

Input has little effect on the worker — with output fixed at 1024, growing input from 128 to 8192 only raises worker time from 17.63s to 19.80s (+12%):

input	worker med (s) @ out=1024
128	17.63
512	17.71
1024	17.83
2048	18.14
4096	18.76
8192	19.80

The pure prefill cost can be read approximately from the output=1 rows (input=8192 / out=1 → 0.886s).

Conclusion: worker cost $\approx$ output_len $\times$ 17.5 ms, consistent with decode being serial and memory-bandwidth-bound.

3. Verifier Cost Model: Total-Dominated, Parallel Prefill

The verifier is a single prefill; its time scales linearly with total (input+output):

total	verifier med (s)	throughput (tok/s)
129	0.033	(includes fixed overhead)
1152	0.170	~6800
8320	1.488	~5600
16384	3.129	~5240

Conclusion: verifier cost $\approx$ total_tokens / ~5200 + fixed overhead. Prefill computes all positions in one parallel pass, making it two orders of magnitude faster than serial decode.

4. Key Metric: Worker/Verifier Cost Ratio

This is the most valuable column of the experiment. It grows sharply with output before flattening to a constant. Representative scenarios:

Scenario (input/output)	worker gen (s)	verifier check (s)	ratio
512 / 1024	17.71	0.217	81.5×
128 / 2048	35.40	0.312	113.6×
2048 / 4096	73.04	0.819	89.2×
1024 / 4096	72.24	0.691	104.6×

The full ratio matrix by output length (across input):

output →	1	16	64	256	1024	2048	4096	8192
input=128	0.82	8.36	26.27	70.10	103.78	113.56	84.45	97.99
input=512	0.65	3.67	13.20	39.35	81.51	99.84	76.71	97.24
input=1024	0.64	2.47	7.33	25.02	62.45	84.70	104.62	92.44
input=2048	0.70	1.59	4.39	14.60	42.78	64.15	89.18	82.83
input=4096	0.78	1.20	1.90	8.33	19.58	46.38	51.76	62.43
input=8192	0.62	0.81	1.40	3.88	12.19	21.16	36.65	50.13

How to read the table:

- **The output=1 column is entirely < 1:** the verifier is slightly slower than the worker due to the per-position logprobs overhead.
- **Each row rises monotonically with output:** the longer the output, the more cost-effective verification becomes.
- **Each column decreases as input grows:** with longer input, the verifier must pay for the extra input-segment prefill, narrowing the ratio.
- **For a fixed input the ratio rises then dips slightly** (e.g. input=128 peaks at 113.6 at out=2048, falling back to 98 at out=8192): when output $\gg$ input, the verifier's total $\approx$ output also scales $\propto$ output, so the ratio asymptotes to $17.5\text{ms} / \sim 0.19\text{ms} \approx 90\text{--}100\times$.

5. Notes and Limitations

- **The TTFT / decode split columns are empty (-).** This run used the vLLM V1 engine (the `EngineCore` in the logs), which does not expose the legacy `RequestMetrics.first_token_time`, so the script cannot separate TTFT from decode. This does not affect the total timings or any conclusion above; for a prefill approximation, use the output=1 rows.
- **This is single-request latency at batch=1.** In real online serving, the worker amortizes per-token cost via continuous batching, and the verifier can verify in batches, so absolute numbers will change. However, the fundamental difference between prefill (parallel) and decode (serial) remains, so the order-of-magnitude relationship — "verification is far cheaper than generation" — still holds.

6. Reproduction

```
cd vllm_worker_verify_pipeline
./benchmark_worker_verifier_latency.py \
  --model Qwen/Qwen3-8B \
  --dtype bfloat16 --trust-remote-code \
  --max-model-len 8192 \
  --input-lengths "128 512 1024 2048 4096 8192" \
  --output-lengths "1 16 64 256 1024 2048 4096 8192" \
  --top-k 20 --warmup 2 --repeats 5 \
  --gpu "4090" \
  --output-dir results/worker_verifier_latency/qwen3_8b_4090
```

Artifacts: `latency_summary.csv` (aggregated metrics), `latency_raw.jsonl` (per-iteration raw timings), `summary.md` (Markdown table), `metadata.json` (args and environment).

Appendix: Full Data Table

input	output	total	worker med (s)	verifier med (s)	worker/verifier
128	1	129	0.0272	0.0332	0.82
128	16	144	0.2839	0.0340	8.36
128	64	192	1.1085	0.0422	26.27
128	256	384	4.4061	0.0629	70.10
128	1024	1152	17.6304	0.1699	103.78
128	2048	2176	35.3973	0.3117	113.56
128	4096	4224	71.4902	0.8465	84.45
128	8192	8320	145.7669	1.4876	97.99
512	1	513	0.0549	0.0849	0.65
512	16	528	0.3137	0.0854	3.67
512	64	576	1.1387	0.0863	13.20
512	256	768	4.4452	0.1130	39.35
512	1024	1536	17.7100	0.2173	81.51
512	2048	2560	35.5538	0.3561	99.84
512	4096	4608	71.8107	0.9361	76.71
512	8192	8704	146.3028	1.5046	97.24
1024	1	1025	0.1016	0.1586	0.64
1024	16	1040	0.3624	0.1467	2.47
1024	64	1088	1.1905	0.1625	7.33
1024	256	1280	4.5075	0.1802	25.02
1024	1024	2048	17.8318	0.2855	62.45
1024	2048	3072	35.7824	0.4224	84.70
1024	4096	5120	72.2362	0.6905	104.62
1024	8192	9216	147.0083	1.5903	92.44
2048	1	2049	0.2040	0.2910	0.70
2048	16	2064	0.4668	0.2937	1.59
2048	64	2112	1.3059	0.2973	4.39
2048	256	2304	4.6611	0.3193	14.60
2048	1024	3072	18.1371	0.4239	42.78
2048	2048	4096	36.2336	0.5648	64.15
2048	4096	6144	73.0383	0.8190	89.18
2048	8192	10240	148.4488	1.7922	82.83

input	output	total	worker med (s)	verifier med (s)	worker/verifier
4096	1	4097	0.4181	0.5344	0.78
4096	16	4112	0.6856	0.5693	1.20
4096	64	4160	1.5416	0.8107	1.90
4096	256	4352	4.9817	0.5982	8.33
4096	1024	5120	18.7645	0.9585	19.58
4096	2048	6144	37.2070	0.8023	46.38
4096	4096	8192	74.6061	1.4413	51.76
4096	8192	12288	151.2620	2.4228	62.43
8192	1	8193	0.8863	1.4333	0.62
8192	16	8208	1.1655	1.4470	0.81
8192	64	8256	2.0467	1.4572	1.40
8192	256	8448	5.5912	1.4422	3.88
8192	1024	9216	19.7963	1.6246	12.19
8192	2048	10240	38.8759	1.8373	21.16
8192	4096	12288	77.5542	2.1159	36.65
8192	8192	16384	156.8459	3.1287	50.13

[Compare with the earlier Qwen2.5-7B verification-cost experiment →](#)

[Back to papers and experiments →](#)