Network Working Group
Internet-Draft
Intended status: Informational
Expires: 12 October 2026

M. Mondal
M. Gaikwad
Independent

10 April 2026

Benchmarking Workload Profiles for Large Language Model Serving
draft-mondal-llm-serving-workload-profiles-00

Abstract

This document defines standard workload profiles for benchmarking Large Language Model (LLM) inference serving systems. Each profile represents one class of production workload. A profile is defined by its input and output token distribution, output structure, latency sensitivity, concurrency pattern, and caching behavior. Profiles are organized into six groups: Non-Generative, Minimal Output, Interactive Streaming, Prefill-Heavy Generative, Decode-Heavy Generative, and Multi-Step Chained. This document works with "Benchmarking Methodology for Large Language Model Serving" [LLM-METHOD]. It specifies the workloads that the methodology’s tests SHOULD be run against.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 12 October 2026.

Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction

The companion document [LLM-METHOD] defines test procedures, measurement specifications, and reporting formats for LLM inference systems. A second companion [LLM-TERMS] defines the metrics vocabulary. Together these two documents specify how to measure. This document specifies what to measure against.

Production LLM deployments serve many different task types. These tasks differ in fundamental inference properties.

Some workloads have no decode phase at all. Embedding workloads are one example.

Some workloads produce free-form text. Others produce output constrained to a schema or a label set.

Some users wait interactively for streaming tokens. Others consume results in batch.

Some requests share a large input prefix with other requests. Others arrive independently.

Some workloads are a single inference call. Others are a chain of dependent calls.

These properties determine which metrics from [LLM-TERMS] are meaningful. They determine which tests from [LLM-METHOD] apply. They determine what performance characteristics matter in production.

Without standard profiles, benchmark results are hard to compare. A system tuned for chat may perform poorly on summarization. A system benchmarked only on synthetic uniform workloads may not match any real production scenario. This document fixes that.

1.1. Relationship to Companion Documents

This document is the third in a series.

[LLM-TERMS] defines the metrics vocabulary. Examples are Time to First Token, Inter-Token Latency, and Output Token Throughput.

[LLM-METHOD] defines test procedures. Examples are how to measure TTFT, how to set up load generators, and how to format reports.

This document defines workload profiles. It specifies what representative tasks to benchmark against. It maps each profile to the applicable metrics and tests.

Implementers SHOULD select profiles that match their production workload mix. They SHOULD apply the tests from [LLM-METHOD] to those profiles. They SHOULD report results per profile.

1.2. Document Structure

Section 3 defines the classification framework. It uses five dimensions to describe each profile. Section 4 defines the common benchmark run parameters that MUST be reported for all profiles (Section 4.1). Sections 4.2 through 4.7 define the profiles. Section 5 maps profiles to applicable metrics. Section 6 covers mixed-workload composition. Section 7 covers dimensions that apply to any profile. Section 9 covers IANA considerations.

1.3. Out of Scope

The following topics are outside the scope of this document.

Model training is out of scope. This document covers inference serving only.

Model quality is out of scope. Output correctness and accuracy are not performance metrics here. The one exception is output validity rates for structured output profiles, which MAY be reported as a secondary metric.

Network and client-side latency are out of scope. The exception is when the system under test (SUT) boundary is defined to include network transport per [LLM-METHOD].

Multimodal workloads are out of scope. This document covers text-only LLM serving. Multimodal input and output such as images, audio, and video are left for future work.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Profile Classification Framework

Each profile is described along five dimensions. These dimensions allow systematic comparison across profiles. They help implementers understand the inference properties of each workload.

3.1. Input/Output Ratio Class

This ratio is computed as input tokens divided by output tokens. It determines the balance between the prefill phase and the decode phase. It is often the main factor in whether a workload is compute-bound or memory-bandwidth-bound. The exact boundary depends on model architecture, batch size, quantization, and attention implementation.

For profiles that produce hidden thinking tokens (see Section 4.6.2), those tokens count as output tokens for the ratio. The exception is when the serving system hides them entirely from the caller. In that case, the serving system MUST document whether max_output_tokens applies to visible tokens only, to total tokens, or to both.

Prefill-Only (PO): No token generation. The output is a vector, a score, or a probability distribution. Examples are embedding, cross-encoder reranking, and perplexity scoring.

Prefill-Dominant (PD): Input is much larger than output. The ratio is greater than 10:1. Examples are summarization, classification, and extraction.

Balanced (BA): Input and output are comparable in length. The ratio is between 1:3 and 3:1. Examples are translation, conversational chat, and rewriting.

Decode-Dominant (DD): Output is much larger than input. The ratio is greater than 1:10. Examples are content generation, creative writing, and data generation.

3.2. Output Constraint Level

This dimension describes how much the output format is constrained. It affects decode efficiency, guided generation overhead, and output validation requirements.

Non-Token (NT): The output is not a token sequence. Examples are vectors, floating-point scores, and probability distributions.

Minimal (MI): The output is a single token, a short label, or a numeric score. Examples are classification and yes/no decisions.

Structured (ST): The output MUST conform to a defined schema. Examples are JSON objects, SQL queries, and function call arguments.

Semi-Structured (SS): The output follows a general format but has variable content. Examples are Markdown, numbered lists, and tables.

Free-Form (FF): There are no structural constraints. Examples are natural language text, creative writing, and conversational responses.

3.3. Latency Sensitivity Class

This dimension describes how much latency the consumer can tolerate. It determines whether TTFT, ITL, or throughput is the primary optimization target.

The latency targets listed below are informative. They reflect targets commonly used in production. They are not normative benchmark pass/fail thresholds. Actual targets depend on model size, hardware, network configuration, and operator SLOs.

Interactive (IN): The user watches tokens arrive in real time. Interactive deployments often target sub-500ms TTFT as a user-experience goal. ITL consistency is critical. Examples are chat and code completion.

Near-Real-Time (NR): The response is needed within seconds. It is usually not streamed token by token. Production deployments commonly target total latency below 5 seconds. Examples are search ranking, extraction, and function calling.

Throughput-Oriented (TO): Latency is secondary. The goal is to maximize aggregate throughput. Examples are batch classification, bulk summarization, and data generation.

3.4. Concurrency Pattern

This dimension describes the typical request arrival and batching pattern. It affects scheduler behavior and resource utilization.

Single-Stream (SI): Requests arrive individually. They follow a Poisson or similar arrival process. Examples are general API usage and chat.

Burst-Batch (BB): Groups of related requests arrive together. Examples are ranking a set of candidates and classifying a document batch.

Sustained-High (SH): A continuous high-volume stream of requests arrives at a stable rate. Examples are production pipelines and data processing.

Chained-Sequential (CS): Each request depends on the output of the previous request. Examples are agentic workflows and planning pipelines.

3.5. Prefix Sharing Pattern

This dimension describes how much requests share common input prefixes. It affects KV cache utilization and memory efficiency.

In this document, prefix sharing means cross-request shared prefill reuse. A serving system retains and reuses the computed key-value (KV) state for a shared input prefix across multiple distinct requests. This is different from two other things that are sometimes called caching.

The first is intra-request KV cache. This is the normal decode-phase cache retained within a single request. It is present in all generative profiles. It is not what this dimension measures.

The second is external result caching. This is deduplication of identical complete requests at an API gateway or proxy layer. This is out of scope for this document.

When a profile has a non-None prefix sharing value, benchmarks MUST report prefix cache hit rates. A cache hit means the serving system determined that the leading N tokens of an incoming request match a cached prefix exactly. This match is token-by- token or via an equivalent hash. The cached KV state is then reused without recomputation. Serving systems MUST document the cache eviction policy and the cache type as part of the run parameters defined in Section 4.1.

None (N): Each request has a unique prefix. Example is independent short queries.

System-Prompt (S): A shared system prompt appears across requests. It is typically 100 to 2000 tokens. Most API deployments follow this pattern.

Document-Shared (D): Multiple requests share a large document context. It is typically thousands to tens of thousands of tokens. Examples are ranking candidates against a shared passage and multi-question QA over a document.

Turn-Accumulated (T): The prefix grows across conversation turns. Examples are multi-turn chat and agentic tool loops.

Schema-Shared (H): Requests share a tool or function schema as a prefix. Examples are function calling and SQL generation with a shared database schema.

4. Profile Groups and Definitions

This section defines 25 workload profiles in six groups. Section 4.1 specifies common benchmark run parameters that apply to all profiles. Sections 4.2 through 4.7 define the profiles.

Each profile includes the following.

Description: What the task is and why it is a distinct benchmarking profile.

Classification Vector: The five-dimensional classification from Section 3.

Typical Token Distribution: Representative input and output token length ranges.

Key Performance Indicators: The most important metrics from [LLM-TERMS] for this profile.

Benchmarking Considerations: Profile-specific setup, measurement, or reporting guidance.

4.1. Common Benchmark Run Parameters

Benchmark results are only comparable when the configuration is fully specified. The following parameters MUST be reported for every benchmark run. They apply in addition to any profile- specific requirements in Sections 4.2 through 4.7.

Parameter	Requirement
Model identifier and version	MUST report exact model name and version string
Tokenizer name and version	MUST report; tokenizer affects all token count measurements
Decoding strategy	MUST report: greedy, sampling, or beam search
Temperature	MUST report if sampling; N/A for greedy
top_p	MUST report if sampling
top_k	MUST report if sampling
max_output_tokens	MUST report; specify whether limit applies to visible tokens, total tokens, or both
Stop sequences	MUST report all stop conditions used
Streaming enabled	MUST report: enabled or disabled
Server token flush policy	MUST report if streaming enabled (e.g., per-token, batched-N, time-based)
Guided decoding enabled	MUST report: enabled or disabled
Guided decoding method	MUST report if enabled (e.g., grammar-based, JSON schema, logit masking)
Guided decoding schema size	MUST report if enabled: schema token count and nesting depth
Prefix cache	MUST report: enabled or disabled

enabled	
Prefix cache type	MUST report if enabled (e.g., block-level KV cache, full-prefix cache)
Prefix cache eviction policy	MUST report if enabled (e.g., LRU, TTL-based, capacity limit)
Hardware configuration	MUST report: accelerator type, count, and memory capacity
Serving framework and version	MUST report
Quantization	MUST report: precision (e.g., fp16, int8, int4) and method if applicable

Table 1

Implementers SHOULD also report the version of any load generation tool used. They SHOULD report the network topology between the load generator and the SUT.

4.2. Group A: Non-Generative Profiles

Non-generative profiles produce no output token sequence. The inference pipeline runs only the prefill pass. It returns a non-token result. This result is a vector embedding, a scalar score, or a per-token probability distribution.

There is no decode phase. The following metrics from [LLM-TERMS] do not apply to Group A profiles.

* Time to First Token (TTFT)

* Inter-Token Latency (ITL)

* Output Token Throughput

* Time Per Output Token (TPOT)

Group A profiles MUST instead be measured using the following.

Request Latency: Total time from request submission to result receipt.

Sequences Per Second: Number of complete inference requests

processed per second under the specified concurrency. For Group A profiles, one sequence equals one request. Seq/s and Req/s are the same here. When a single API call submits multiple sequences in a batch, throughput MUST be reported as sequences per second, not API calls per second.

Prefill Throughput: Input tokens processed per second.

4.2.1. EMBED: Embedding Generation

The model encodes input text into a fixed-size dense vector. This vector is used for semantic search, retrieval, clustering, and similarity computation. This is one of the highest-volume LLM inference workloads in production. Every RAG pipeline, recommendation system, and semantic search index depends on embedding generation.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Only (PO)
Output Constraint	Non-Token (NT)
Latency Sensitivity	Near-Real-Time (NR) to Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	None (N)

Table 2

Typical Token Distribution:

Parameter	Range
Input Length	16 to 8192 tokens
Output	Fixed-size vector (e.g., 768, 1024, 1536, or 3072 dimensions)

Table 3

Key Performance Indicators:

* Sequences per second at target batch size

* P50 and P99 request latency under load

* Prefill throughput in input tokens per second

* Throughput scaling with batch size

Embedding workloads work well with large batches. Benchmarks SHOULD measure throughput across batch sizes of 1, 8, 32, 64, 128, and 256. This shows batching efficiency. Input length SHOULD come from a representative corpus. Real embedding workloads have high length variance because short queries and long passages are both common. Benchmarks SHOULD report latency and throughput separately for query-length inputs (fewer than 64 tokens) and passage-length inputs (256 to 8192 tokens). These two cases often serve different stages of a retrieval pipeline.

4.2.2. XRANK: Cross-Encoder Reranking

The model receives a query and a document. It produces a relevance score without generating tokens. Cross- encoder reranking is used in search pipelines to refine initial retrieval results. The model processes the concatenated query and document pair. It outputs a scalar relevance score.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Only (PO)
Output Constraint	Non-Token (NT)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Burst-Batch (BB)
Prefix Sharing	Document-Shared (D)

Table 4

Typical Token Distribution:

Parameter	Range
Input Length	128 to 1024 tokens (query plus document)
Output	Single scalar score

Table 5

Key Performance Indicators:

* Pairs scored per second at target batch size

* P50 and P99 request latency per scoring batch

* Throughput scaling as candidate set size grows

Cross-encoder reranking typically processes a batch of candidates. This is commonly 10 to 100 query-document pairs. The query is shared across all pairs. Benchmarks SHOULD measure performance with the shared query prefix to capture KV cache benefits. Candidate set sizes of 10, 25, 50, and 100 SHOULD be tested. Latency SHOULD be reported for the full candidate set. This is the time to score all candidates, not just one pair. This matches real search pipeline requirements.

4.2.3. LOGPR: Logprob / Perplexity Scoring

The model processes input text. It returns per- token log probabilities or a sequence-level perplexity score. It does not generate new tokens. This is used for AI-generated content detection, model evaluation, data quality filtering, watermark detection, and calibration. The model runs a forward pass over the input. It returns the probability distribution at each token position.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Only (PO)
Output Constraint	Non-Token (NT)
Latency Sensitivity	Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	None (N)

Table 6

Typical Token Distribution:

Parameter	Range
Input Length	64 to 4096 tokens
Output	Per-token logprobs or aggregate perplexity score

Table 7

Key Performance Indicators:

* Sequences scored per second

* Input tokens processed per second

* P50 and P99 request latency by input length bucket

Logprob scoring is computationally the same as the prefill phase of generative inference. The difference is that it returns probability data instead of starting a decode phase. It differs from embedding in one key way. The full vocabulary probability distribution may be returned at each position. This creates large output data volume even though no tokens are generated. Benchmarks SHOULD specify whether full vocabulary logprobs or only top-k logprobs are returned. This choice significantly affects data transfer overhead. Input length SHOULD span the full range to show prefill scaling.

4.3. Group B: Minimal Output Profiles

Minimal output profiles generate a small number of tokens. The output is a label, a score, a short answer, or a small structured object. The decode phase is short relative to prefill. These workloads are typically prefill-dominant and work well with batching.

For Group B profiles, the metric priorities differ from streaming workloads.

Total Request Latency is the primary latency metric. TTFT and ITL are not primary here.

Request Throughput in requests per second is more useful than output token throughput.

TTFT is still measurable. It is approximately equal to total latency because the decode phase is very short.

4.3.1. CLAS: Classification and Labeling

The model reads input text. It produces a categorical label or set of labels from a predefined taxonomy. Examples are sentiment analysis, content moderation, topic categorization, intent detection, and spam filtering.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Minimal (MI)
Latency Sensitivity	Near-Real-Time (NR) to Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	System-Prompt (S)

Table 8

Typical Token Distribution:

Parameter	Range
Input Length	32 to 2048 tokens
Output Length	1 to 10 tokens

Table 9

Key Performance Indicators:

* Requests classified per second

* P50 and P99 total request latency

* Throughput scaling with batch size

Classification workloads often use constrained decoding or guided generation to restrict output to valid labels. Benchmarks SHOULD report whether guided generation is enabled. They SHOULD report which method is used, such as grammar-based or logit masking. System prompts typically include the label taxonomy. They SHOULD be included in the benchmark setup. The label set size SHOULD be specified. Binary, multi-class, and multi-label variants have different guided generation overhead.

4.3.2. SFQA: Short-Form QA / Factoid

The model receives a short question. It may also receive brief context. It produces a concise factual answer. Examples are customer support FAQ, knowledge base queries, and simple information retrieval. It differs from long-document QA (LDQA) in that the input context is short.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA) to Prefill-Dominant (PD)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Interactive (IN) to Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	System-Prompt (S)

Table 10

Typical Token Distribution:

Parameter	Range
Input Length	16 to 512 tokens
Output Length	5 to 100 tokens

Table 11

Key Performance Indicators:

* TTFT if streamed

* Total request latency

* Requests per second under concurrency

This profile represents the simple API call pattern. It is the baseline for many LLM applications. It SHOULD be used as a reference workload to establish baseline system performance before testing more complex profiles. Input length SHOULD include very short queries (fewer than 32 tokens) and context- augmented queries (256 to 512 tokens).

4.3.3. RANK: Generative Ranking / Scoring

The model evaluates one or more candidates. It produces a relevance score, a preference ordering, or a comparative judgment as generated tokens. It differs from cross- encoder reranking (XRANK) in that it generates token output such as scores, explanations, or ordinal labels. XRANK produces a raw scalar from the model's hidden state.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Structured (ST) to Minimal (MI)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Burst-Batch (BB)
Prefix Sharing	Document-Shared (D)

Table 12

Typical Token Distribution:

Parameter	Range
Input Length	256 to 4096 tokens (prompt plus candidates)
Output Length	5 to 50 tokens (scores or ranking)

Table 13

Key Performance Indicators:

* Time to score the full candidate set

* Requests per second

* Prefix cache hit rate across candidates sharing context

Ranking workloads have strong prefix sharing. When ranking N candidates against a shared query or document, the shared prefix may be 80 to 90 percent of total input tokens. Benchmarks MUST report prefix cache hit rates. They SHOULD compare performance with and without prefix caching. Candidate set sizes of 5, 10, 20, and 50 SHOULD be tested.

4.3.4. NER: Entity Resolution and Recognition

The model reads input text. It identifies named entities and produces a structured list of entity mentions with their types and positions. This is used for information extraction from documents, log parsing, and data enrichment pipelines.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Structured (ST)
Latency Sensitivity	Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	System-Prompt (S)

Table 14

Typical Token Distribution:

Parameter	Range
Input Length	64 to 2048 tokens
Output Length	20 to 200 tokens (structured entity list)

Table 15

Key Performance Indicators:

* Documents processed per second

* Total request latency by input length

* Throughput under batch processing

NER output length varies with input content density. Benchmarks SHOULD use representative documents with known entity density. This ensures output length distributions are realistic. Guided generation to a JSON schema is common. Whether it is enabled SHOULD be reported.

4.3.5. FUNC: Function Calling / Tool Selection

The model receives a user request and a set of tool or function definitions. It selects the right function and produces well-formed arguments. The tool schema definitions form a large and highly cacheable prefix.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Structured (ST)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	Schema-Shared (H)

Table 16

Typical Token Distribution:

Parameter	Range
Input Length	512 to 4096 tokens (schema plus user query)
Output Length	20 to 200 tokens (function name plus JSON arguments)

Table 17

Key Performance Indicators:

* Total request latency

* TTFT as an indicator of schema processing time

* Prefix cache hit rate for shared tool schemas

Tool schemas vary widely in size. A small schema with 5 tools may be a few hundred tokens. A large schema with 50 tools and complex parameters may be several thousand tokens. Benchmarks SHOULD test with schema sizes of approximately 500, 2000, and 4000 tokens. Tool schemas are highly shareable as prefixes. Prefix caching effectiveness is a critical measurement. Constrained decoding to valid JSON is standard and SHOULD be enabled.

4.3.6. SQLN: SQL / Query Generation

The model receives a natural language question and a database schema definition. It produces a syntactically valid SQL query. The schema prefix is large and shared across queries against the same database.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Structured (ST)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI) to Burst-Batch (BB)
Prefix Sharing	Schema-Shared (H)

Table 18

Typical Token Distribution:

Parameter	Range
Input Length	256 to 8192 tokens (schema plus question)
Output Length	20 to 300 tokens (SQL query)

Table 19

Key Performance Indicators:

* Total request latency

* Prefix cache effectiveness for shared schema

* Requests per second under concurrent users on the same schema

Database schemas range from simple to complex. A simple schema may have 5 tables and around 500 tokens. A complex schema may have 100 or more tables and 8000 or more tokens. Benchmarks SHOULD test with at least two schema sizes. Multiple queries against the same schema are the dominant production pattern. Prefix caching measurements are critical for this profile.

4.4. Group C: Interactive Streaming Profiles

Interactive streaming profiles serve users who watch tokens arrive in real time. These profiles share four key properties.

TTFT is critical for perceived responsiveness. ITL consistency determines perceived fluency. Multi-turn context accumulation is common. Prefix caching across conversation turns adds value.

For Group C profiles, all streaming metrics from [LLM-TERMS] apply. These are TTFT, ITL, TPOT, and output token throughput. The Throughput-Latency Tradeoff test from [LLM-METHOD] is especially important for this group.

4.4.1. CHAT: Conversational Chat

A user and the model have a multi-turn conversation. Each turn adds to the conversation history. This creates a growing input context. This is the most common interactive LLM workload. It is the default profile for general- purpose chat applications.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Interactive (IN)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	Turn-Accumulated (T)

Table 20

Typical Token Distribution:

Parameter	Range
Input Length	64 to 16384 tokens (grows with turns)
Output Length	50 to 1000 tokens
Turns per conversation	3 to 20

Table 21

Key Performance Indicators:

* TTFT at various context lengths

* P50 and P99 ITL

* TTFT degradation as conversation grows

* Prefix cache hit rate across turns

Chat benchmarks MUST test across multiple conversation depths. This shows how TTFT scales as context grows. Tests at turns 1, 5, 10, and 20 SHOULD be included at minimum. Input data SHOULD come from real multi-turn conversation datasets such as ShareGPT [SHAREGPT] or LMSYS-Chat [LMSYS]. This captures realistic length distributions. The time between turns, while the user reads and types, affects cache eviction pressure. This time SHOULD be set as a parameter in the benchmark.

4.4.2. COMP: Code Completion / Autocomplete

The model receives a code context. This includes the file content, cursor position, and surrounding code. It generates a short completion. This profile has very tight latency requirements, very short output, and high request frequency. Prefix sharing is strong because the same file context appears in many consecutive requests.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD) to Balanced (BA)
Output Constraint	Semi-Structured (SS)
Latency Sensitivity	Interactive (IN)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	Document-Shared (D)

Table 22

Typical Token Distribution:

Parameter	Range
Input Length	256 to 8192 tokens (file context)
Output Length	5 to 200 tokens (completion)

Table 23

Key Performance Indicators:

* TTFT (primary). Interactive deployments often target sub-200ms TTFT. Benchmarks SHOULD report TTFT at P50, P95, and P99.

* Total completion latency

* Prefix cache hit rate (consecutive edits share most context)

Code completion has the tightest TTFT requirements of any profile. Benchmarks SHOULD simulate realistic editing patterns. In these patterns, consecutive requests share 95 percent or more of input tokens with small modifications. Fill-in-the-middle (FIM) variants SHOULD be tested if the serving system supports it. In FIM, the model generates tokens to fill a gap within existing code rather than appending to the end. Both single-line and multi-line completions SHOULD be measured.

4.4.3. PERS: Dialogue / Roleplay / Persona

The model maintains a persistent character or persona defined by a large system prompt. This differs from generic chat. System prompts may be 2000 to 10000 tokens or more. They include character descriptions, world-building context, and behavioral guidelines. System prompt prefix caching is important when many users share the same persona.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Interactive (IN)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	System-Prompt (S) plus Turn-Accumulated (T)

Table 24

Typical Token Distribution:

Parameter	Range
System Prompt	2000 to 10000 tokens
User Turn Input	16 to 256 tokens
Output Length	100 to 2000 tokens

Table 25

Key Performance Indicators:

* TTFT with large system prompts

* System prompt prefix cache hit rate across concurrent users

* ITL consistency during long responses

The key benchmark for this profile is performance under large shared system prompts with many concurrent users. When 1000 users share the same persona, the system prompt KV cache should be computed once. Benchmarks SHOULD measure TTFT and throughput as a function of the number of concurrent users sharing the same system prompt. Results with and without prefix caching SHOULD both be reported.

4.5. Group D: Prefill-Heavy Generative Profiles

Prefill-heavy profiles process long input contexts. They generate moderate-length output. The prefill phase dominates compute and memory requirements. These workloads stress KV cache capacity, long-context attention efficiency, and memory management.

The most important metrics for Group D profiles are TTFT (heavily influenced by prefill duration), prefill throughput in input tokens per second, memory utilization at peak context length, and total request latency.

4.5.1. SUMM: Summarization

The model reads a document or set of passages. It produces a condensed summary. This is one of the most common production LLM tasks. The input is often 10 to 100 times longer than the output. The prefill phase dominates total latency.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Free-Form (FF) to Semi-Structured (SS)
Latency Sensitivity	Near-Real-Time (NR) to Throughput-Oriented (TO)
Concurrency Pattern	Single-Stream (SI) to Sustained-High (SH)
Prefix Sharing	None (N)

Table 26

Typical Token Distribution:

Parameter	Range
Input Length	1024 to 32768 tokens
Output Length	64 to 1024 tokens

Table 27

Key Performance Indicators:

* TTFT by input length bucket

* Total request latency

* Prefill throughput in input tokens per second

* Documents summarized per second

Summarization performance depends strongly on input length. Benchmarks MUST report results in input length buckets. Example buckets are 1K to 4K, 4K to 8K, 8K to 16K, and 16K to 32K tokens. Input data SHOULD come from document corpora with realistic length distributions rather than synthetic text. Both extractive summarization (shorter output) and abstractive summarization (longer output) SHOULD be tested.

4.5.2. LDQA: Long Document Question Answering

The model receives a long document and a question. It produces a concise answer from the document content. It differs from summarization in that the output is shorter and more targeted. It differs from short-form QA in that it has a large context document.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI) to Burst-Batch (BB)
Prefix Sharing	Document-Shared (D)

Table 28

Typical Token Distribution:

Parameter	Range
Input Length	4096 to 131072 tokens
Output Length	16 to 512 tokens

Table 29

Key Performance Indicators:

* TTFT at various context lengths
* TTFT scaling as context length doubles
* Total request latency
* Prefix cache effectiveness across multiple questions

Long document QA tests the interaction between context length and latency. Benchmarks SHOULD test with multiple questions over the same document. This measures prefix caching effectiveness. Context lengths SHOULD span at least three orders of magnitude. Example values are 4K, 16K, 64K, and 128K tokens. The position of the answer within the document can affect performance on some systems. This MAY be reported.

4.5.3. MDOC: Multi-Document Synthesis

The model receives multiple documents. It produces one output that synthesizes information across all of them. Examples are literature review, competitive analysis, and multi-source report generation. It differs from single-document summarization in that it requires reasoning across documents. It differs from RAG in that there is no retrieval step. All documents are provided in full.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Semi-Structured (SS)
Latency Sensitivity	Throughput-Oriented (TO)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	None (N)

Table 30

Typical Token Distribution:

Parameter	Range
Input Length	8192 to 131072 tokens (multiple documents)
Output Length	256 to 4096 tokens

Table 31

Key Performance Indicators:

* TTFT at extreme context lengths

* Total request latency

* Memory utilization at peak context

* Output throughput during the decode phase

Multi-document synthesis pushes context length to its maximum. Benchmarks SHOULD report the number of documents and the per-document length separately. This shows whether performance depends on total token count alone or also on document structure. This profile is a stress test for memory management and long-context attention implementations.

4.5.4. EXTR: Structured Extraction

The model reads an input document. It extracts specific information into a defined output schema. Examples are invoice parsing, resume extraction, form processing, and log analysis. The output MUST conform to a specified JSON schema, XML structure, or tabular format.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Structured (ST)
Latency Sensitivity	Near-Real-Time (NR) to Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	System-Prompt (S)

Table 32

Typical Token Distribution:

Parameter	Range
Input Length	256 to 16384 tokens
Output Length	50 to 500 tokens (structured JSON or XML)

Table 33

Key Performance Indicators:

* Documents processed per second

* Total request latency

* Guided generation overhead vs. an unconstrained baseline

Extraction workloads typically use constrained or guided decoding to ensure the output conforms to the schema. Benchmarks MUST report whether guided generation is enabled. They MUST report the schema complexity including the number of fields and nesting depth. Benchmarks SHOULD compare throughput with and without guided generation. This shows the overhead. Output validation rates, meaning the percentage of outputs that conform to the schema, SHOULD be reported alongside performance metrics.

4.5.5. JUDG: LLM-as-Judge / Evaluation

The model evaluates another model’s output against specified criteria. It produces a score and optionally a justification. This is used in production for quality assurance, A/B testing, content moderation review, and RLHF reward modeling. The input includes the original prompt, the response to evaluate, and the scoring criteria.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD)
Output Constraint	Structured (ST) to Semi-Structured (SS)
Latency Sensitivity	Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	System-Prompt (S)

Table 34

Typical Token Distribution:

Parameter	Range
Input Length	512 to 8192 tokens (criteria plus prompt plus response)
Output Length	10 to 300 tokens (score plus reasoning)

Table 35

Key Performance Indicators:

* Evaluations per second

* Total request latency

* Throughput under sustained batch load

LLM-as-Judge is a high-volume pipeline workload. Benchmarks SHOULD simulate evaluation of a corpus of responses. They SHOULD measure sustained throughput over thousands of evaluations. The scoring criteria form a cacheable system prompt prefix. Both score-only mode (minimal output) and score-with-reasoning mode (longer output) SHOULD be tested.

4.5.6. RAGN: Retrieval-Augmented Generation

The model receives retrieved context passages and a user query. It generates a grounded response. It differs from other prefill-heavy profiles in that the context is made of multiple retrieved chunks assembled together. It differs from LDQA in that the input context is shorter and is made of separately retrieved passages rather than a single contiguous document.

Classification Vector:

Dimension	Value
I/O Ratio	Prefill-Dominant (PD) to Balanced (BA)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Interactive (IN) to Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	System-Prompt (S)

Table 36

Typical Token Distribution:

Parameter	Range
Retrieved Context	512 to 8192 tokens (3 to 20 chunks)
User Query	16 to 256 tokens
Total Input	528 to 8448 tokens
Output Length	64 to 1024 tokens

Table 37

Key Performance Indicators:

* TTFT (the user is typically waiting interactively)

* Total request latency including time to first useful content

* End-to-end latency including retrieval if measured at the compound system boundary

RAG benchmarks SHOULD vary the number and size of retrieved chunks. This shows how performance scales as context grows. Configurations of 3, 5, 10, and 20 retrieved chunks SHOULD be tested. The SUT boundary from [LLM-METHOD] is especially important for RAG. Measuring at the model engine boundary excludes retrieval latency. Measuring at the compound system boundary includes it. Both SHOULD be reported when the retrieval system is part of the SUT.

4.6. Group E: Decode-Heavy Generative Profiles

Decode-heavy profiles generate much more output than input. The decode phase dominates total latency and compute. These workloads stress memory bandwidth, ITL consistency, and sustained output token throughput.

For Group E profiles, output token throughput and ITL distribution are the primary metrics. TTFT is less important relative to total generation time.

4.6.1. CGEN: Content Generation

The model receives a short prompt or specification. It generates long-form content. Examples are blog posts, articles, marketing copy, emails, reports, and creative writing. Output is much longer than input and is free-form.

Classification Vector:

Dimension	Value
I/O Ratio	Decode-Dominant (DD)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Interactive (IN) to Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	System-Prompt (S)

Table 38

Typical Token Distribution:

Parameter	Range
Input Length	32 to 1024 tokens
Output Length	256 to 8192 tokens

Table 39

Key Performance Indicators:

* Output token throughput in tokens per second

* P50 and P99 ITL

* ITL consistency over long generation (stutter detection)

* Total generation time

Long-form generation is the main test of sustained decode performance. Benchmarks SHOULD measure ITL distribution across the full generation. Measuring only the initial tokens is not sufficient. Decode performance can degrade as the KV cache grows. Plotting ITL against output position is useful for detecting decode slowdown. Output lengths of 256, 1024, 2048, and 4096 or more tokens SHOULD be tested.

4.6.2. REAS: Reasoning / Chain-of-Thought

The model performs extended step-by-step reasoning before producing a final answer. This profile has grown in importance with reasoning-optimized models that produce explicit or hidden thinking tokens. The input is short. The output is very long, often 10 to 100 times the length of the final answer. Most of the output is intermediate reasoning steps.

Classification Vector:

Dimension	Value
I/O Ratio	Decode-Dominant (DD)
Output Constraint	Free-Form (FF) to Semi-Structured (SS)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	System-Prompt (S)

Table 40

Typical Token Distribution:

Parameter	Range
Input Length	64 to 2048 tokens
Reasoning Output	512 to 32768 tokens (thinking or CoT)
Final Answer	32 to 512 tokens
Total Output	544 to 33280 tokens

Table 41

Key Performance Indicators:

* Total generation time (may be 30 to 120 seconds or more)

* Output token throughput sustained over long generation

* ITL consistency over extended decode

* Time to final answer if thinking tokens are hidden

Reasoning workloads produce the longest output sequences of any profile. This creates unique stress on KV cache management during decode. Hidden thinking tokens are a serving-system- specific feature. Benchmarks MUST distinguish between visible tokens and hidden thinking tokens if the serving system supports this. Serving systems that hide thinking tokens MUST document whether

max_output_tokens applies to visible tokens only, to total tokens, or to both. This MUST be reported in the run parameters defined in Section 4.1. Time to first visible token, which comes after reasoning completes, is a meaningful metric. It differs from standard TTFT and SHOULD be reported when thinking tokens are hidden. Benchmarks SHOULD test with reasoning budgets of 1K, 4K, 16K, and 32K tokens. Memory pressure during extended reasoning is a critical measurement.

4.6.3. DGEN: Data Generation / Synthetic Data

The model receives a schema or specification. It generates multiple structured records. This is used for test data generation, training data augmentation, simulation, and content pipelines. The output is repetitive, structured, and produced at high volume.

Classification Vector:

Dimension	Value
I/O Ratio	Decode-Dominant (DD)
Output Constraint	Structured (ST)
Latency Sensitivity	Throughput-Oriented (TO)
Concurrency Pattern	Sustained-High (SH)
Prefix Sharing	System-Prompt (S)

Table 42

Typical Token Distribution:

Parameter	Range
Input Length	128 to 2048 tokens (schema plus instructions)
Output Length	512 to 16384 tokens (multiple records)

Table 43

Key Performance Indicators:

* Output tokens per second (sustained)

* Records generated per second

* Throughput under high concurrency

Data generation is a throughput-oriented profile. TTFT and ITL are not primary metrics here. Benchmarks SHOULD measure sustained output throughput over large generation batches of 10K or more records. Request latency is still measurable and MAY be reported as a secondary metric. Guided generation to a JSON or CSV schema is standard and SHOULD be enabled. Schema validation rate SHOULD be reported.

4.6.4. TRNS: Translation

The model translates text from one natural language to another. The input-to-output ratio is approximately 1:1, varying by language pair. The full input content is consumed in producing the output.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Near-Real-Time (NR) to Throughput-Oriented (TO)
Concurrency Pattern	Single-Stream (SI) to Sustained-High (SH)
Prefix Sharing	System-Prompt (S)

Table 44

Typical Token Distribution:

Parameter	Range
Input Length	32 to 8192 tokens
Output Length	32 to 10000 tokens
I/O Ratio	0.5:1 to 2:1 depending on language pair

Table 45

Key Performance Indicators:

* Total request latency by segment length

* Output tokens per second

* Throughput in segments translated per second

Token-to-token ratio varies by language pair because tokenizer coverage differs across languages. Benchmarks MUST specify the language pair and tokenizer used. CJK languages typically produce more tokens per semantic unit than European languages. Benchmarks SHOULD test with at least two language pairs. This shows tokenizer effects. Segment lengths SHOULD span sentence- level (32 to 64 tokens), paragraph-level (128 to 512 tokens), and document-level (2000 or more tokens).

4.6.5. EDIT: Rewriting / Editing

The model receives input text. It produces a transformed version of the same text. Examples are grammar correction, tone adjustment, style transfer, simplification, and paraphrasing. It differs from translation in that it works within the same language. It differs from summarization in that the output length is similar to the input length.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA)
Output Constraint	Free-Form (FF)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Single-Stream (SI)
Prefix Sharing	System-Prompt (S)

Table 46

Typical Token Distribution:

Parameter	Range
Input Length	64 to 4096 tokens
Output Length	64 to 4096 tokens (similar to input)

Table 47

Key Performance Indicators:

* Total request latency

* Output token throughput

* TTFT if streamed to the user for review

Rewriting produces output structurally similar to input. Some serving systems may use this similarity to enable speculative decoding or draft-based approaches. Benchmarks SHOULD report whether such optimizations are enabled.

4.7. Group F: Multi-Step Chained Profiles

Multi-step profiles involve sequences of dependent inference calls. Each call depends on the output of the previous call. Total user-perceived latency is the sum of all step latencies. These workloads test end-to-end pipeline performance rather than single-request optimization.

For Group F profiles, the primary metric is end-to-end pipeline latency. Individual step TTFT and total latency contribute to this but are not sufficient on their own.

4.7.1. AGNT: Agentic / Tool Use

The model operates in a loop. It receives an observation, decides on an action, receives the tool result, and continues until the task is complete. Each iteration is a separate inference call. The context grows across iterations as prior observations and actions accumulate.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA) per step
Output Constraint	Structured (ST) per tool call; Free-Form (FF) for final answer
Latency Sensitivity	Near-Real-Time (NR) to Interactive (IN)
Concurrency Pattern	Chained-Sequential (CS)
Prefix Sharing	Turn-Accumulated (T)

Table 48

Typical Token Distribution (per step):

Parameter	Range
Input Length	512 to 16384 tokens (grows with steps)
Output Length	20 to 500 tokens per step
Steps per task	3 to 20

Table 49

Key Performance Indicators:

* End-to-end task completion time

* Per-step TTFT (compounds across steps)

* Per-step total latency

* TTFT scaling as context accumulates across steps

* Prefix cache hit rate across steps

Agentic workloads are especially sensitive to per-step TTFT. Latencies compound across steps. A 500ms TTFT per step over 10 steps adds 5 seconds of TTFT alone to total task completion time. Benchmarks MUST simulate realistic multi-step trajectories. They MUST include tool call outputs injected between steps. Context grows across steps, so prefix caching across turns is critical. Benchmarks SHOULD test with task lengths of 3, 5, 10, and 20 steps. Both successful paths and long paths SHOULD be tested.

4.7.2. PLAN: Planning / Task Decomposition

The model receives a high-level goal. It produces a structured plan with ordered steps, dependencies, and resource requirements. This is often the first step of an agentic pipeline. It may involve iterative refinement where the model reviews and revises its own plan.

Classification Vector:

Dimension	Value
I/O Ratio	Balanced (BA) to Decode-Dominant (DD)
Output Constraint	Semi-Structured (SS)
Latency Sensitivity	Near-Real-Time (NR)
Concurrency Pattern	Chained-Sequential (CS)
Prefix Sharing	Turn-Accumulated (T)

Table 50

Typical Token Distribution:

Parameter	Range
Input Length	128 to 4096 tokens (goal plus context)
Output Length	256 to 2048 tokens (structured plan)
Refinement Rounds	1 to 3

Table 51

Key Performance Indicators:

* Total planning time including refinement rounds

* Per-round latency

* Output token throughput

Planning with iterative refinement requires the model to evaluate and modify its own prior output. Benchmarks SHOULD test both single-pass planning and multi-round refinement. Plan quality is outside the scope of performance benchmarking. The time to produce a plan of a given length and structure is measurable and SHOULD be reported.

5. Profile-to-Metric Applicability Matrix

The matrix below maps each profile to the applicable metrics from [LLM-TERMS]. "Primary" means the most important metric for that profile. "Applicable" means it is meaningful but secondary. "-" means it does not apply.

TTFT and ITL are streaming-only metrics. They apply only when streaming is enabled per Section 7.2. When streaming is disabled, only total request latency is measurable.

For Group F profiles, "Request Latency" means end-to-end pipeline latency across all steps. "TTFT" means per-step TTFT, which compounds across the pipeline.

Request latency is still measurable in batch mode but is secondary to throughput metrics.

Sub-table 1 lists applicability for TTFT, ITL, TPOT, and output token throughput.

Profile	TTFT	ITL	TPOT	Output Tok/s
Group A: Non-Generative				
EMBED	-	-	-	-
XRANK	-	-	-	-
LOGPR	-	-	-	-
Group B: Minimal Output				
CLAS	Applicable	-	-	-
SFQA	Primary	Applicable	-	-
RANK	Applicable	-	-	-
NER	Applicable	-	Applicable	-
FUNC	Primary	Applicable	-	-
SQLN	Applicable	-	-	-
Group C: Interactive				
CHAT	Primary	Primary	Applicable	Applicable
COMP	Primary	Applicable	-	-
PERS	Primary	Primary	Applicable	Applicable
Group D: Prefill-Heavy				
SUMM	Primary	Applicable	Applicable	Applicable
LDQA	Primary	Applicable	-	-
MDOC	Primary	Applicable	Applicable	Applicable
EXTR	Applicable	-	Applicable	Applicable
JUDG	Applicable	-	Applicable	-
RAGN	Primary	Primary	Applicable	Applicable
Group E: Decode-Heavy				

CGEN	Applicable	Primary	Primary	Primary
REAS	Applicable	Primary	Primary	Primary
DGEN	-	-	Applicable	Primary
TRNS	Applicable	Applicable	Primary	Primary
EDIT	Applicable	Applicable	Primary	Applicable
Group F: Multi-Step				
AGNT	Primary	Applicable	-	-
PLAN	Applicable	Applicable	Applicable	Applicable

Table 52

Sub-table 2 lists applicability for request latency, request throughput, sequence throughput, and prefill throughput.

Profile	Req Latency	Req/s	Seq/s	Prefill Tok/s
Group A: Non-Generative				
EMBED	Primary	Primary	Primary	Applicable
XRANK	Primary	Primary	Primary	Applicable
LOGPR	Primary	Applicable	Primary	Primary
Group B: Minimal Output				
CLAS	Primary	Primary	-	Applicable
SFQA	Primary	Applicable	-	-
RANK	Primary	Primary	-	Applicable
NER	Primary	Primary	-	Applicable
FUNC	Primary	Applicable	-	Applicable
SQLN	Primary	Applicable	-	Applicable

Group C: Interactive				
CHAT	Applicable	-	-	-
COMP	Primary	-	-	Applicable
PERS	Applicable	-	-	Applicable
Group D: Prefill-Heavy				
SUMM	Primary	Applicable	-	Primary
LDQA	Primary	Applicable	-	Primary
MDOC	Primary	-	-	Primary
EXTR	Primary	Primary	-	Primary
JUDG	Primary	Primary	-	Applicable
RAGN	Primary	-	-	Applicable
Group E: Decode-Heavy				
CGEN	Applicable	-	-	-
REAS	Primary	-	-	-
DGEN	Applicable	Primary	-	-
TRNS	Primary	Applicable	-	-
EDIT	Primary	-	-	-
Group F: Multi-Step				
AGNT	Primary	-	-	Applicable
PLAN	Primary	-	-	-

Table 53

6. Profile Composition for Mixed Workloads

Production LLM deployments rarely serve a single profile. A typical deployment might serve 60 percent CHAT, 20 percent RAGN, 10 percent SUMM, and 10 percent FUNC traffic at the same time. Benchmarking with mixed profiles is needed to understand real- world performance.

6.1. Weighted Profile Combinations

A mixed workload specification MUST include the following: the set of profiles included; the traffic weight for each profile as a percentage of total requests or total tokens; and whether weights are in request count or token volume.

Example mixed workload specification:

Profile	Request Weight	Token Weight (approx.)
CHAT	60%	45%
RAGN	20%	30%
SUMM	10%	20%
FUNC	10%	5%

Table 54

Request weight and token weight differ because profiles have different average token counts per request. Both SHOULD be reported.

6.2. Reporting Requirements for Mixed Workloads

When benchmarking mixed workloads, the following apply.

Overall aggregate metrics including total throughput and aggregate latency distribution MUST be reported.

Per-profile metrics MUST be reported separately. A mixed workload benchmark that reports only aggregate metrics hides profile-specific performance characteristics.

The interaction between profiles MUST be described. For example, long-running REAS requests may increase queuing delay for latency-sensitive CHAT requests under continuous batching.

Scheduling fairness metrics from [LLM-METHOD] SHOULD be applied to mixed workloads. This shows whether any profile has worse latency than expected.

The benchmark run parameters from Section 4.1 MUST be reported for the full mixed workload run. When parameters differ across profiles within the same run, the per-profile overrides MUST be reported.

7. Cross-Cutting Dimensions

The following dimensions apply to any profile in Section 4. They describe deployment or configuration choices. They are not task characteristics.

7.1. Batch vs. Online

Any profile can be run in batch mode or online mode. Batch mode processes a queue of requests with throughput as the priority. Online mode serves interactive requests with latency as the priority. The profile definition gives the typical latency sensitivity. Deployers MAY run any profile in either mode.

Benchmark reports MUST specify the deployment mode.

Online mode uses open-loop load generation with a specified arrival rate. Latency metrics are primary.

Batch mode uses closed-loop operation with maximum concurrency. Throughput metrics are primary.

7.2. Streaming vs. Non-Streaming

Generative profiles in Groups B through F can be served with or without streaming. Streaming delivers tokens as they are generated. Non-streaming returns the complete response at once.

With streaming, TTFT and ITL are measurable and meaningful. Without streaming, only total request latency is measurable.

The streaming mode MUST be reported. Profiles with Interactive latency sensitivity (Section 3.3) SHOULD be tested with streaming enabled. Where the matrix in Section 5 marks TTFT or ITL as Primary or Applicable, those metrics apply only when streaming is enabled.

7.3. Fill-in-the-Middle

Fill-in-the-middle (FIM) is a variant that applies primarily to the COMP profile. In FIM, the model generates tokens to fill a gap within existing text rather than appending to the end. FIM changes the prefill pattern because the model receives both a prefix and a suffix context. FIM is not a separate profile.

When a profile is tested in FIM mode, this MUST be reported. FIM-capable serving systems may show different prefill performance because of the bidirectional context.

8. Security Considerations

8.1. Workload-Specific Attack Surfaces

Different profiles have different security implications.

Decode-heavy profiles in Group E can be used as resource exhaustion vectors by requesting maximum output lengths. Benchmark setups MUST enforce output length limits consistent with the production configuration.

Multi-step profiles in Group F may accumulate context across steps. Benchmark configurations MUST specify maximum steps and total context limits.

Structured output profiles may be vulnerable to schema injection attacks. These cause excessive guided generation backtracking. Benchmarks SHOULD use representative schemas rather than adversarially crafted ones.

8.2. Profile Gaming

Systems may be tuned for specific profiles at the expense of others. Single-profile benchmarks can be misleading. Benchmark reports SHOULD include results across multiple profiles that represent the intended production workload mix.

8.3. Data Privacy in Reference Datasets

Reference datasets for benchmarking (Appendix A) SHOULD NOT contain personally identifiable information (PII). When using conversation datasets such as those for the CHAT profile, data MUST be anonymized or synthetic.

9. IANA Considerations

This document has no IANA actions.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[LLM-TERMS] Gaikwad, M., "Benchmarking Terminology for Large Language Model Serving", Work in Progress, Internet-Draft, draft-gaikwad-llm-benchmarking-terminology-00, January 2026, <https://datatracker.ietf.org/doc/html/draft-gaikwad-llm-benchmarking-terminology-00>.

[LLM-METHOD] Gaikwad, M., "Benchmarking Methodology for Large Language Model Serving", Work in Progress, Internet-Draft, draft-gaikwad-llm-benchmarking-methodology-00, January 2026, <https://datatracker.ietf.org/doc/html/draft-gaikwad-llm-benchmarking-methodology-00>.

10.2. Informative References

[SHAREGPT] "ShareGPT52K", 2023, <https://huggingface.co/datasets/RyokoAI/ShareGPT52K>.

[LMSYS] Zheng, L., Chiang, W., Sheng, Y., Li, T., and et al., "LMSYS-Chat-1M: A Large-Scale Real-World LLM Conversation Dataset", arXiv 2309.11998, September 2023, <https://arxiv.org/abs/2309.11998>.

Appendix A. Reference Datasets for Each Profile

This appendix lists recommended datasets for building workloads for each profile. Implementers MAY use other datasets. They MUST report the dataset used and its token length statistics.

Implementers are responsible for checking dataset license terms before use. The datasets listed here are referenced for their technical relevance. No license terms are implied or endorsed. The PII requirements in Section 8.3 apply regardless of which dataset is chosen.

For the PERS profile, the recommended source is synthetic persona datasets. Implementers MUST describe the synthetic dataset used. At minimum they MUST report the system prompt token length distribution (mean, P50, P90, P99), the vocabulary and domain coverage, and the generation method or tool used.

Profile	Recommended Dataset(s)	Notes
EMBED	MTEB benchmark passages; MS MARCO passages	Query and passage length variants
XRANK	MS MARCO query-passage pairs	Candidate sets of 10 to 100
LOGPR	OpenWebText; The Pile subsets	Representative web text
CLAS	SST-2; AG News; GoEmotions	Binary, multi-class, multi-label variants
SFQA	Natural Questions; TriviaQA	Short-context factoid questions
RANK	MS MARCO; TREC-DL	Relevance judgment tasks
NER	CoNLL-2003; OntoNotes	Standard NER benchmarks
FUNC	Berkeley Function Calling Leaderboard	Varied tool schema complexity
SQLN	Spider; WikiSQL	Simple and complex schema variants
CHAT	ShareGPT; LMSYS-Chat-1M	Real multi-turn conversations; MUST be anonymized
COMP	The Stack; HumanEval	Real code files with cursor positions

PERS	Synthetic persona datasets	Large system prompts (2K to 10K tokens); see characterization requirements above
SUMM	CNN/DailyMail; arXiv; GovReport	Short and long document variants
LDQA	NarrativeQA; QuALITY; Scrolls	Long-document QA benchmarks
MDOC	Multi-XScience; WikiSum	Multi-source synthesis
EXTR	SQuAD extractive; CORD; SROIE	Schema-constrained extraction
JUDG	MT-Bench judgments; Arena-Hard	LLM evaluation datasets
RAGN	Natural Questions plus retriever output	Realistic retrieved chunk assembly
CGEN	WritingPrompts; Alpaca instructions	Creative and instructional generation
REAS	GSM8K; MATH; ARC-Challenge	Problems requiring extended reasoning
DGEN	Spider schema definitions; synthetically generated JSON schemas	Schema-directed generation; implementers MUST document schema generation method
TRNS	WMT translation benchmarks	Multiple language pairs
EDIT	JFLEG; IteraTeR	Grammar correction and style transfer
AGNT	SWE-bench; WebArena trajectories	Multi-step tool use traces
PLAN	ALFWorld; HotpotQA decompositions	Planning and decomposition tasks

Table 55

Appendix B. Profile Parameter Quick Reference Table

Sub-table 1 lists the profile identity, group, I/O ratio, and output constraint.

Profile	Code	Group	I/O Ratio	Output Constraint
Embedding Generation	EMBED	A	PO	NT
Cross-Encoder Reranking	XRANK	A	PO	NT
Logprob Scoring	LOGPR	A	PO	NT
Classification	CLAS	B	PD	MI
Short-Form QA	SFQA	B	BA-PD	FF
Generative Ranking	RANK	B	PD	ST-MI
Entity Recognition	NER	B	PD	ST
Function Calling	FUNC	B	PD	ST
SQL Generation	SQLN	B	PD	ST
Conversational Chat	CHAT	C	BA	FF
Code Completion	COMP	C	PD-BA	SS
Persona Dialogue	PERS	C	BA	FF
Summarization	SUMM	D	PD	FF-SS
Long Document QA	LDQA	D	PD	FF
Multi-Doc Synthesis	MDOC	D	PD	SS

Structured Extraction	EXTR	D	PD	ST
LLM-as-Judge	JUDG	D	PD	ST-SS
RAG	RAGN	D	PD-BA	FF
Content Generation	CGEN	E	DD	FF
Reasoning / CoT	REAS	E	DD	FF-SS
Data Generation	DGEN	E	DD	ST
Translation	TRNS	E	BA	FF
Rewriting / Editing	EDIT	E	BA	FF
Agentic / Tool Use	AGNT	F	BA/step	ST+FF
Planning	PLAN	F	BA-DD	SS

Table 56

Sub-table 2 repeats the profile key and lists latency class, concurrency, prefix sharing, and token ranges.

Profile	Latency Class	Concurrency	Prefix Sharing	Input Tokens	Output Tokens
EMBED	NR-TO	SH	N	16-8K	N/A (vector)
XRANK	NR	BB	D	128-1K	N/A (score)
LOGPR	TO	SH	N	64-4K	N/A (probs)
CLAS	NR-TO	SH	S	32-2K	1-10

SFQA	IN-NR	SI	S	16-512	5-100
RANK	NR	BB	D	256-4K	5-50
NER	TO	SH	S	64-2K	20-200
FUNC	NR	SI	H	512-4K	20-200
SQLN	NR	SI-BB	H	256-8K	20-300
CHAT	IN	SI	T	64-16K	50-1K
COMP	IN	SI	D	256-8K	5-200
PERS	IN	SI	S+T	2K-10K sys	100-2K
SUMM	NR-TO	SI-SH	N	1K-32K	64-1K
LDQA	NR	SI-BB	D	4K-128K	16-512
MDOC	TO	SI	N	8K-128K	256-4K
EXTR	NR-TO	SH	S	256-16K	50-500
JUDG	TO	SH	S	512-8K	10-300
RAGN	IN-NR	SI	S	528-8K	64-1K
CGEN	IN-NR	SI	S	32-1K	256-8K
REAS	NR	SI	S	64-2K	544-33K
DGEN	TO	SH	S	128-2K	512-16K
TRNS	NR-TO	SI-SH	S	32-8K	32-10K
EDIT	NR	SI	S	64-4K	64-4K
AGNT	NR-IN	CS	T	512-16K	20-500/ step
PLAN	NR	CS	T	128-4K	256-2K

Table 57

Acknowledgements

The authors thank the IETF BMWG community for their foundational work in benchmarking methodology.

Authors' Addresses

Mohadeb Mondal
Independent
Email: mohadeb.mondal@gmail.com

Madhava Gaikwad
Independent
Email: gaikwad.madhav@gmail.com