THE AI TOOLCHAIN

NO. 014

Head

THE WEEK'S TOP PICKS

PUBLISHED NOVEMBER 20, 2026 · FRIDAYS

// SHARE THIS WEEK ▽ Top 5 ▽ Top Pick

EDITIONS tail | grep | **head** | ls | diff | uniq | perf

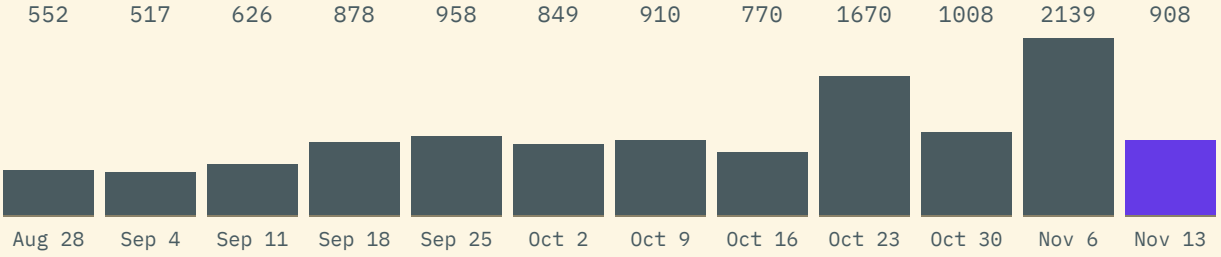
The best tooling updates that shipped this week.

The week in numbers · Nov 13–19

908	321	4	578
changes we read →	tools shipped →	product launches →	breaking changes →

Changes per week, last 12 weeks · Nov 13–19

Changes we read in that week's dailies, including docs and API changes. A gap means no daily was published.



Category moves this month

Plays could not be checked. This does not mean there is nothing to do. Retry

vim ? help

▸ // HOW THESE PICKS ARE MADE SHOW

▾ // BENCHMARKS MOVED FOR TOOLS WE TRACK HIDE

Gemini API APEX-Agents Gemini 4 Neon `high`

82.7% +4.7 pts released 2026-11-18 · leaderboard ↗

OpenAI APEX-Agents GPT-6.2 Sol `max`

66.6% −1.7 pts released 2026-11-17 · leaderboard ↗

Gemini API APEX-Agents Gemini 4 Neon `xhigh`

82.9% +4.9 pts released 2026-11-18 · leaderboard ↗

OpenAI APEX-Agents GPT-6.2 Astra `max`

72.2% +3.9 pts released 2026-11-17 · leaderboard ↗

OpenAI Codex CLI Terminal-Bench 4.0 GPT-6.2 Astra `max`

62.1% +8.8 pts listed 2026-11-17 · leaderboard ↗

OpenAI Codex CLI Terminal-Bench 4.0 GPT-6.2 Sol `max`

61.5% +8.2 pts listed 2026-11-17 · leaderboard ↗

Gemini API APEX-Agents Gemini 4 Hydrogen

78.0% −6.2 pts released 2026-11-12 · leaderboard ↗

Claude Code Terminal-Bench 4.0 Hy4 Preview `max`

19.1% −46.7 pts listed 2026-08-28 · leaderboard ↗

Claude Code Terminal-Bench 4.0 Kimi K3.1 Pro `max`

59.7% −6.1 pts listed 2026-10-24 · leaderboard ↗

OpenAI Codex CLI Terminal-Bench 4.0 GPT-5.4 Mini `xhigh`

1.8% −59.1 pts listed 2026-03-17 · leaderboard ↗

Claude Code Terminal-Bench 4.0 Mistral Large 4 `max`

27.6% −38.2 pts listed 2026-10-06 · leaderboard ↗

OpenAI Codex CLI Terminal-Bench 4.0 GPT-6.1 Terra `max`

53.3% −7.6 pts listed 2026-10-26 · leaderboard ↗

Scores from Epoch AI Benchmarking Hub (CC BY 4.0), evaluated 2026-11-13 to 2026-11-20.

▾ // LOCAL LLM HARDWARE THIS WEEK HIDE

Apple Mac Studio measured

Qwen3.6-35B-A3B-4bit 4bit · 132.18 tok/s generate on M5 Ultra (llama.cpp 0.32.2)

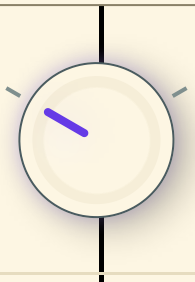
2026-11-13 · source ↗

Vendor list prices, captured live and via the Wayback Machine; MSRP for cards until street capture ships. Events dated 2026-11-13 to 2026-11-20; each device page carries the full history.

VIEW

ISSUE VIEW

scroll



SCROLL | EXPAND IN PLACE

INFORMATION DENSITY

everything



BRIEFED | EVERYTHING

Every change in this week's picks, in full.

all 30 changes

Do you prefer **scroll**? 👍 👎

This week's 5

01	**TensorRT-LLM**	Feature · Inference Serving & Optimization
02	**ONNX Runtime**	Feature · Inference Serving & Optimization
03	**Amazon Bedrock AgentCore**	Feature · ML Platforms & SDKs
04	**TRL**	Feature · Model Training & Fine-Tuning
05	**LM Studio**	Feature · Local LLM Runtimes

The picks in full

01 TensorRT-LLM v1.3.0

Inference Serving & Optimization

TensorRT-LLM compiles and serves large language models on NVIDIA GPUs with quantization, in-flight batching, and speculative decoding.

// WHY IT MADE THE LIST — DEPTH 5/5 — IMPACT 5/5 — NOVELTY 4/5

TensorRT-LLM v1.3.0 adds a VisualGen vertical for image and video generation (FLUX, Wan, HunyuanVideo, MiniMax-H3, Cosmos3) alongside LLM inference, plus an Anthropic Messages API endpoint, a gRPC server, and MXFP8 quantization — a single release that opens three previously unavailable serving categories on NVIDIA hardware.

> TensorRT-LLM v1.3.0 adds a VisualGen vertical for image and video generation, Anthropic Messages and embeddings API endpoints, new MXFP8 and NVFP4 weight-only quantization paths, and a gRPC server, while completing its move to a PyTorch-only backend by removing the legacy TensorRT backend, C++ Executor API and related sampler and speculative-decoding code paths.

└──▷ WHAT SHIPPED · 9 FEATURES most completely described first ? what's the number?

01 PyTorch-only backend; TensorRT backend and legacy APIs removed

Not checked yet — BREAKING — 85

The TensorRT backend is removed; PyTorch is the only backend, removing `trtllm-build`, `TrtLlmArgs`, `BuildConfig`, `MedusaDecodingConfig`, `LookaheadDecodingConfig`, `trtllm-bench build` and `trtllm-bench --engine_dir`. The legacy C++ `Executor` API, legacy runtime and batch-manager APIs, and the `cpp_only` build option are also removed; TensorRT-engine Triton backend models are removed so only the `llmapi` Triton model remains; `TRTLLMSampler` is removed in favor of `TorchSampler` (requiring FlashInfer, with `sampler_type` ignored and `disable_flashinfer_sampling` removed); two-model speculative decoding is removed, dropping the `eagle3_one_model` and `mtp_eagle_one_model` fields in favor of one-model implementations; and AutoDeploy, Star Attention and WIDEEP are removed.

— Enumerates every removed API, flag and migration path v1.3.0

02 New serving endpoints and frontends Not checked yet NEW

80

Adds new serving endpoints and frontends: OpenAI-compatible `/v1/messages` for Anthropic Messages API clients and `/v1/embeddings` with an llm.encode() fast path

for encoder-only model embeddings, Prometheus metrics with new `/start_profile` and `/stop_profile` endpoints for profiling control, and a new gRPC server and multi-process HTTP frontends.

Obtain text embeddings from an encoder-only model for downstream retrieval or reranking.

```python
from tensorrt_llm import LLM

llm = LLM(model="nvidia/NV-Embed-v2")
embeddings = llm.encode(["Represent this sentence for retrieval:", "TensorRT-LLM is fast."])
print(embeddings[0].outputs.embedding)
```

— Named endpoints with a runnable code example Named endpoints but no example or metric detail Bare mention with no flags or example

v1.3.0

---

03 **New trtllm-serve CLI flags** Not checked yet NEW 80

Adds new `trtllm-serve` options `--served_model_name`, `--kv_cache_dtype` and `--middleware` to control model aliasing, KV cache precision and request middleware.

Serve a model with a custom alias and FP8 KV cache to control memory use and match a client-facing name.

```
$ trtllm-serve "nvidia/Llama-3.1-8B-Instruct-FP8" --served_model_name my-llama --kv_cache_dtype fp8
```

— Exact flags demonstrated with a runnable serve command v1.3.0

---

04 **MXFP8 and W4A16 NVFP4 weight-only quantization** Not checked yet NEW 75

Adds MXFP8 (W8A8) linear and MoE quantization, including loading MXFP8 checkpoints such as `MiniMax-M3`, and a new W4A16 NVFP4 weight-only quantization path through Marlin to run NVFP4 checkpoints on Hopper and Ada Lovelace.

— Names formats, checkpoints and GPU targets, no command v1.3.0

---

05 **VisualGen image/video generation vertical** Not checked yet NEW 70

Adds a new VisualGen vertical to `trtllm-serve` for image and video generation alongside LLM inference, supporting `FLUX.1`, `FLUX.2`, GLM-Image, Qwen-Image,

`Wan 2.2`, `LTX-2`, `HunyuanVideo 1.5`, `Cosmos3` and `MiniMax-H3`, plus a new `trtllm-eval` pipeline to evaluate VisualGen models.

— Names nine supported models and eval tool, no exact command `v1.3.0`

06 **Locality domain control for NVFP4 MoE on Rubin** Not checked yet

NEW

65

Control locality domain (uGPU) localization for NVFP4 MoE on Rubin — enabled by default; disable with `DISABLE_LOCALITY_DOMAINS=1`.

— Named env var and default behaviour given `v1.3.0`

07 **Request-level priority, tool choice and logprobs** Not checked yet

NEW

60

Adds per-request `priority`, named-function `tool_choice`, `parallel_tool_calls` and logprobs support to the OpenAI-compatible APIs, including the Completions API.

— Names fields but gives no usage example `v1.3.0`

THINNER COVERAGE BELOW

08 **RL workflow support with routing replay and inflight updates**

Not checked yet NEW

50

Adds RL workflow support with Router Replay's `return_routed_experts` field for returning per-token MoE routing decisions, and inflight weight updates for updating model weights in-flight.

— Names field but no usage example or workflow steps `v1.3.0`

09 **Experimental reasoning token budget cap** Not checked yet NEW

40

Adds an experimental `thinking_token_budget` parameter to cap reasoning-model token budgets.

— Named parameter but no usage context `v1.3.0`

└──▷ **BREAKING ON UPGRADE**

! The TensorRT backend is removed; PyTorch is the only backend. `trtllm-build`, `TrtLlmArgs`, `BuildConfig`, `MedusaDecodingConfig`, `LookaheadDecodingConfig`, `trtllm-bench build` and `trtllm-bench --engine_dir` are removed.

! The legacy C++ `Executor` API, the legacy runtime and batch-manager APIs, and the `cpp_only` build option are removed.

! TensorRT-engine Triton backend models are removed; only the `llmapi` Triton model remains.

! `TRTLLMSampler` is removed; `TorchSampler` (requiring FlashInfer) is the only sampler. The `sampler_type` argument is ignored with a warning, and `disable_flashinfer_sampling` is removed.

! Two-model speculative decoding is removed; the `eagle3_one_model` and `mtp_eagle_one_model` fields are removed — use the one-model implementations.

! AutoDeploy, Star Attention and WIDEEP are removed.

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

02 ONNX Runtime plugin-ep-cuda/v0.4.0

Inference Serving & Optimization

ONNX Runtime: cross-platform, high performance ML inferencing and training accelerator

// WHY IT MADE THE LIST DEPTH 5/5 IMPACT 5/5 NOVELTY 4/5

The ONNX Runtime CUDA Plugin EP now offloads MoE experts between CPU and GPU memory, letting practitioners serve Mixture-of-Experts models that exceed GPU VRAM — a workload that was previously off the table without multi-node setups — with popularity-guided budgeting and configurable memory limits per session.

ONNX Runtime CUDA Plugin EP adds CPU/GPU MoE expert offloading, native GGUF block-packed weight operators, and INT2/INT3/INT4 Tensor-Core QMoE kernels for large-model inference.

└──▷ GET THIS VERSION ▾

```
$ git clone --branch plugin-ep-cuda/v0.4.0
https://github.com/microsoft/onnxruntime.git
# already have the repo? check out this version:
$ git checkout plugin-ep-cuda/v0.4.0
```

Session-scoped CPU/GPU hybrid expert offloading for `com.microsoft::QMoE` nodes, configured via `session.moe.offload.*` options with no model changes required. Not checked yet

New `GatherPacked`, `MatMulPacked`, and `MoEPacked` operators consume GGUF block-packed quantized weights in native layout without offline repacking. Not checked yet

New offline K-packed NVFP4 QMoE weight path (`weights_prepacked=1`) with register-decoded GEMV and single-CTA fused softmax/top-k routing for up to 512 experts. Not checked yet

NEW SUPPORTED PLATFORM New fast INT3 and mixed-width QMoE kernels with BF16 Tensor-Core decode GEMVs for INT4, INT3, and INT2 experts. Not checked yet

Shared MoE GEMM tactic-profiling cache across identically shaped layers reduces session-creation time for large MoE models. Not checked yet

+1 more

NEW SUPPORTED PLATFORM Linux aarch64 CUDA 13 GPU support

RETIRED SUPPORTED PLATFORM ACL (Arm Compute Library) execution provider removed

RETIRED SUPPORTED PLATFORM In-tree NvTensorRTRTXExecutionProvider provider bridge removed

└──▷ BREAKING ON UPGRADE

- ! Using `session.moe.offload.*`, pipelined external-weight loading, or `OrtApi::EpDevice_GetMemoryCapacity` requires ONNX Runtime core 1.32.0.
- ! Models using `GatherPacked`, `MatMulPacked`, `MoEPacked`, or ONNX opset 28 require ONNX Runtime core 1.32.0.
- ! With ONNX Runtime 1.32.0 core and `session.resource_cuda_partitioning_settings`, `GroupQueryAttention` nodes are declined by the plugin EP.
- ! Runtime fusions `ep.cuda.runtime_fusion.unit_swish` and `ep.cuda.runtime_fusion.causal_conv_with_state` are not available in CUDA plugin builds.
- ! Core and the CUDA plugin must be built from the same ONNX Runtime revision to avoid contributed-operator schema mismatches.

▾ 1 more release this week · 2026-11-13 → 2026-11-14

v1.32.0

ONNX Runtime adds CPU/GPU hybrid MoE offloading, new memory-capacity and weightless-compilation APIs, zero-copy buffer borrowing, and sub-second session init on large graphs.

└──▷ GET THIS VERSION ▾

```
$ git clone --branch v1.32.0
https://github.com/microsoft/onnxruntime.git
# already have the repo? check out this version:
$ git checkout v1.32.0
```

Query device total and available memory with the new `OrtApi::EpDevice_GetMemoryCapacity` (C++ EpDevice::GetMemoryCapacity()) and `OrtEpFactory::GetDeviceMemoryCapacity` APIs, with built-in and plugin CUDA EP support. Not checked yet

Control weightless compilation mode explicitly via `ModelCompilationOptions_SetWeightlessMode` or the `ep.enable_weightless_mode` session option (`'0'` = NONE, `'1'` =

`EXTERNAL_ONLY`, `'2'` = ALL), replacing the deprecated `ModelCompilationOptions_SetWeightlessEnabled` and `ep.enable_weightless` options; compiled EPContext models now record `weightless_mode` and `external_dependencies` in model metadata. Not checked yet

Enable zero-copy CPU buffer borrowing in `AddExternalInitializers` via `session.use_external_initializer_file_buffers_directly="1"` for naturally aligned buffers larger than 127 bytes on little-endian hosts. Not checked yet

Accelerate external initializer loading into GPU memory with lock-free multi-worker pinned-memory staging, configurable via `session.external_data_loader.*` session options. Not checked yet

Configure CPU/GPU MoE expert offloading and memory budgets via `session.moe.offload.*` session options, with an optional kernel dispatch policy via `session.kernel_dispatch_policy` (`auto`, `latency`, `memory`, or `safe`). Not checked yet

+3 more

NEW SUPPORTED PLATFORM ONNX opset 28 operator support

NEW SUPPORTED PLATFORM DLPack v1.3 support for all ORT numeric and boolean tensor types

RETIRED SUPPORTED PLATFORM ACL (Arm Compute Library) execution provider removed

RETIRED SUPPORTED PLATFORM In-tree NvTensorRTRTXExecutionProvider provider bridge removed

└──▷ **BREAKING ON UPGRADE**

! The `ACLExecutionProvider` (Arm Compute Library EP), its build options, and its standalone C registration API have been removed; get_all_providers() no longer lists `ACLExecutionProvider` and applications must remove it from explicit provider lists.

! The in-tree provider-bridge `NvTensorRTRTXExecutionProvider` sources have been removed; callers must migrate to the standalone NVIDIA TensorRT RTX plugin EP registered via `OrtApi::RegisterExecutionProviderLibrary`, and legacy `AppendExecutionProvider` calls now return a deprecation error.

Our best guess

Check this claim Claim history

! The DLPack `is_bool_tensor` low-level override has been removed; `uint8` DLPack metadata now always imports as `uint8`.

Our best guess

Check this claim Claim history

! `OrtCompileApi::ModelCompilationOptions_SetWeightlessEnabled` and the `ep.enable_weightless` / `ep.enable_weightless_ep_context_nodes` session options are deprecated; migrate to `ModelCompilationOptions_SetWeightlessMode` and `ep.enable_weightless_mode`.

Our best guess

Check this claim Claim history

! This is the last release distributed via CocoaPods; future releases will use Swift Package Manager.

Our best guess

Check this claim Claim history

GOOD PICK?  GOOD REASON? 

03 Amazon Bedrock AgentCore

Runtime: Run agents with elevated privileges on Instances (2026-11-19)

ML Platforms & SDKs

Amazon Bedrock AgentCore provides managed agent runtimes, memory, identity, gateways, observability, and evaluation services.

// WHY IT MADE THE LIST

DEPTH 4/5 IMPACT 5/5 NOVELTY 4/5

AgentCore Runtime adds nested KVM virtualization and elevated-privilege execution so agents can build and run containers, install system packages, and change kernel settings inside their own session — a capability that previously required a separate VM or human intervention, now available as a managed AWS service. The same batch adds S3 bucket mounts and an AWS Marketplace connector with x402 payment support.

> Amazon Bedrock AgentCore expanded Runtime with elevated-privilege and virtualization support on Instances, S3 bucket mounts, bigger container images, lifecycle hooks and committed resource configurations for V2 agents, while Gateway gained JWT-based access conditions, the AWS Agent Registry gained skill packaging/sync and multi-mode search, and the platform added free monthly web search queries and organization-wide resource control policies.

└──▷ WHAT SHIPPED · 12 FEATURES most completely described first ? what's the number?

01 S3 bucket mount for Runtime agents

Not checked yet NEW

95

Agents can mount an Amazon S3 bucket using the `s3BucketMount` file system configuration; AgentCore runs Mountpoint for Amazon S3 on your behalf so the container needs no mount client. Including `{ runtimeSessionId}` in the configured prefix isolates each session's files within the bucket. Available on platform version V2 runtimes only.

Configure an S3 bucket mount on a V2 agent runtime so every session writes to its own prefix — no mount client required inside your container.

```json
{
  "fileSystemConfiguration": {
    "s3BucketMount": {
      "bucketName": "my-agent-bucket",
      "prefix": "sessions/{ runtimeSessionId}/",
      "mountPath": "/mnt/s3"
    }
  }
}
```

— Full JSON config example with field names and runtime placeholder

Runtime: Mount your own Amazon S3 bucke…

02

JWT claim condition keys for Gateway invocations

Not checked yet

NEW INTEGRATION NEW

85

`InvokeGateway` requests can be filtered by JWT caller claims using the new `bedrock-agentcore:InboundJwtClaim/*` IAM condition keys in gateway resource policies.

Restrict gateway invocations to callers whose JWT carries a specific claim value — for example, allowing only tokens where the `role` claim equals `agent`.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "bedrock-agentcore:InvokeGateway",
      "Resource": "arn:aws:bedrock-agentcore:<region>:<account-
id>:gateway/<gateway-id>",
      "Condition": {
        "StringEquals": {
          "bedrock-agentcore:InboundJwtClaim/role": "agent"
        }
      }
    }
  ]
}
```

— Full IAM policy JSON example with condition key included

Gateway: JWT claim condition keys for g…

03 **Search types and pagination in Agent Registry** Not checked yet

NEW 75

Agent Registry records can be searched by keyword, semantic, or hybrid mode via `searchType` on the registry search API, with results paged using `nextToken`; submitting an empty query lists all records matching a filter. The MCP endpoint's new `search_discoverable_registry_records_paginated` tool accepts the same `searchType` and `nextToken` options.

— Names API params and MCP tool but no example request

AWS Agent Registry: Search types and se…

04 **Committed resource configurations for V2 agents** Not checked yet

NEW 65

V2 agents now support committed baseline vCPU and memory reservations that can burst up to 4 times the baseline, capped at 8 vCPU and 32 GB of memory per session, billed at a lower rate than default consumption-based pricing.

— Gives concrete limits and pricing note but no config example

Runtime: Resource configurations for V2…

05 **Elevated privileges and virtualization on Instances**

Not checked yet NEW 63

Agents on the Instances compute type can now run with elevated privileges — enabling coding agents to build and run containers, install system packages, and change kernel settings — configured via the privilege mode setting documented under 'Configure the privilege mode for Runtime Instances'. Nested virtualization is also supported, letting agents run their own KVM-based virtual machines and emulators inside a session.

— Names the config location but no exact flag values or command

Runtime: Run virtual machines in agents… Runtime: Run agents with elevated privi…

06 **Skill packages and sync in AWS Agent Registry** Not checked yet

NEW 60

A skill package — a `.zip` archive containing a `SKILL.md` and supporting files — can be attached to a skill record in the AWS Agent Registry so consumers can download a validated package directly, and skill records can be synced from Amazon S3 or GitHub so the registry always reflects the latest published package.

— No example of creating or syncing a skill package

AWS Agent Registry: Skill packages and …

07 **Build warmup lifecycle hook for V2 agents** Not checked yet NEW

60

Lifecycle hooks — endpoints in your container that AgentCore Runtime calls at points in the agent runtime's lifecycle — can now be configured for platform version V2 agents. The build warmup hook lets an agent prepare memory and disk while AgentCore Runtime is preparing it, so sessions cold start faster.

— Describes mechanism but no endpoint name or example

Runtime: Build warmup lifecycle hook fo…

THINNER COVERAGE BELOW

08 **Container images up to 10 GB for V2 agents** Not checked yet

IMPROVED

55

V2 agents now support container images up to 10 GB (compressed), up from 2 GB; images larger than 2 GB require a resource configuration.

— States exact size limits but no config example Runtime: Container images up to 10 GB f…

09 **Free monthly web search queries** Not checked yet NEW

45

Amazon Bedrock AgentCore now includes 5,000 free web search queries each month per AWS account, shared between the Web Search Tool and the Web Search API.

— Automatic entitlement with a concrete number but no action needed

Web search: Free monthly queries (2026-…

10 **Resource control policies for AgentCore resources** Not checked yet

NEW

40

Organization-wide access limits can now be set on Amazon Bedrock AgentCore resources using AWS Organizations resource control policies (RCPs), including restrictions on callers that are not IAM principals in your organization.

— Names the policy type but gives no configuration example

Resource control policies for Amazon Be…

11 **VPC mode support for use1-az6 on V2** Not checked yet

NEW SUPPORTED PLATFORM NEW

30

Amazon Bedrock AgentCore Runtime in VPC mode now supports Availability Zone `use1-az6` in US East (N. Virginia) for agents on platform version V2.

— Single AZ addition with no further mechanism described

Runtime: VPC mode now supports use1-az6…

12 **IAM authentication with remote MCP servers** Not checked yet NEW

30

Agents using the Harness can now connect to IAM-protected remote MCP servers without managing separate credentials for each server.

— No config name or example of the IAM auth setup

Harness: IAM authentication with remote…

> Gateway: Discover and call products from AWS Marketplace with the AWS Marketplace conn
>
> AgentCore Gateway adds a built-in AWS Marketplace connector exposing `SearchListings`, `GetListing`, and `ListFulfillmentOptions` MCP tools with x402 payment support.
>
> NEW INTEGRATION New built-in AWS Marketplace connector lets agents search listings, read listing details, and retrieve callable endpoints and payment details via three MCP tools: `SearchListings`, `GetListing`, and `ListFulfillmentOptions`. Not checked yet
>
> Agents can pay for AWS Marketplace products using x402 payments through the connector. Not checked yet

> Runtime: Platform version V2 in nine more Regions (2026-11-12)
>
> NEW SUPPORTED PLATFORM Amazon Bedrock AgentCore Runtime platform version V2 — nine additional Regions

---

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

04 TRL v1.18.0

Model Training & Fine-Tuning

TRL trains transformer language models with reinforcement learning, covering supervised fine-tuning, reward modeling, PPO, DPO, and GRPO.

// WHY IT MADE THE LIST

DEPTH 4/5 IMPACT 4/5 NOVELTY 4/5

TRL promotes AsyncGRPOTrainer to stable with decoupled generation, tree packing, agent training via HarnessAsyncGRPOTrainer, and LoRA adapter sync — the first stable async GRPO path in a major fine-tuning library, directly enabling on-policy RL training for agentic workloads.

> TRL's v1.18.0 release promotes AsyncGRPOTrainer to a stable top-level trainer with decoupled generation, LoRA adapter weight sync, customizable rollout loops, and new metrics, alongside an extensive breaking cleanup of AsyncGRPOConfig and new AMD/AWS Trainium hardware support.

└──▷ **WHAT SHIPPED · 8 FEATURES** most completely described first ? what's the number?

01 Customizable rollout loop via AsyncRolloutLoop subclassing

Not checked yet

95

BREAKING

Customize rollout behavior by subclassing `AsyncRolloutLoop` (importable from `trl.trainer.async_grpo_trainer`) and overriding `filter_prompt`, `generate_one`, `generate_one_turn`, `compute_advantages`, `setup`, or `close`, then passing the class as `loop_cls` on a trainer subclass; this replaces the removed `rollout_worker=` constructor argument on `AsyncGRPOTrainer`.

Override only the rollout generation step (for example, to add custom filtering) while keeping the rest of the async training loop intact.

```python
from trl import AsyncGRPOTrainer
from trl.trainer.async_grpo_trainer import AsyncRolloutLoop,
Rollout

class MyRolloutLoop(AsyncRolloutLoop):
    async def filter_prompt(self, prompt):
        # custom prompt-level filtering logic
        return await super().filter_prompt(prompt)

class MyAsyncGRPOTrainer(AsyncGRPOTrainer):
    loop_cls = MyRolloutLoop

trainer = MyAsyncGRPOTrainer(
    model="Qwen/Qwen3.5-2B",
    reward_funcs=my_reward_fn,
    train_dataset=dataset,
)
trainer.train()
```

— Named methods, class, and a runnable code example v1.18.0

02 **AsyncGRPOConfig timeout, staleness, and batching cleanup**

Not checked yet 93

BREAKING

Control generation staleness with `max_staleness` (in tokens, replacing the removed `queue_maxsize` backpressure control) and the concurrent rollout ceiling with `max_concurrent_rollouts` (default 512, renamed from `max_inflight_tasks`, no longer derived from `per_device_train_batch_size`). `heartbeat_stale_after_s` is renamed to `rollout_worker_timeout` (same 300s default); `request_timeout` and `weight_sync_timeout` are removed and merged into a single `vllm_server_timeout` config key (default 1800s, replacing the former 240s readiness / 600s request timeouts); `fork_threshold_tokens` is removed since multi-turn rollouts always fork on re-tokenization drift; the `token_budget <= 0` fixed-count batching mode is removed, leaving token-budget batching as the only mode (`None` reads `max_model_len` from the server, or pass a positive integer); `repetition_penalty` is removed because it biased the importance ratio; `remove_unused_columns` now defaults to `False` (setting `True` raises an error). The Harbor example's `--max-inflight` flag is renamed to `--max-concurrent-rollouts`, and the OpenEnv/Harbor environment server's `MAX_CONCURRENT_ENVS` must now be at least `max_concurrent_rollouts + 1`.

— Exhaustively names every renamed, removed, and replacement config key v1.18.0

03 **AsyncGRPOConfig weight sync and sampling controls** Not checked yet

NEW 90

`AsyncGRPOConfig` gains new controls: weight sync is now configured entirely through it (NCCL or LoRA adapter sync) instead of the removed `weight_transfer=` constructor argument, with LoRA adapter sync enabled by starting vLLM with `--enable-lora` so each weight sync sends only the adapter (~160 MB at `r=32`) instead of merged weights; score centering with masked importance sampling is now the default objective via `score_centering=True` (set `score_centering=False` to use GRPO's clipped objective instead); and `top_k` can be used alongside `top_p` or `min_p` to renormalize trainer log-probs over the candidate set vLLM sampled from.

Run async GRPO training with generation on one GPU and training on another, overlapping the two for higher throughput.

```
$ # Start the vLLM server on GPU 0
CUDA_VISIBLE_DEVICES=0 VLLM_SERVER_DEV_MODE=1 vllm serve
Qwen/Qwen3.5-2B \
    --weight-transfer-config '{"backend": "nccl"}' \
    --logprobs-mode processed_logprobs \
    --api-server-count 8 \
    --max-model-len 4096

# Train on GPU 1
CUDA_VISIBLE_DEVICES=1 python train_async_grpo.py
```

— Concrete flag, size numbers, and a runnable server/train command Names the config keys and default but no worked example

v1.18.0

---

04 **AsyncGRPOTrainer promoted to stable top-level trainer**

Not checked yet NEW 70

`AsyncGRPOTrainer` and `AsyncGRPOConfig` are now importable directly from `trl` now that the trainer is stable; the old `trl.experimental.async_grpo` path still works but raises a `FutureWarning` until it is removed in v2.0.0.

— Clear migration path but no mechanism beyond the import change v1.18.0

---

05 **New async GRPO training metrics** Not checked yet NEW 65

Track zero-advantage group filtering via the new `frac_zero_std_groups` metric (groups whose rewards are all equal are dropped before the forward and backward pass), plus new async-only metrics including `num_tokens`, `completions/{mean,min,max}_length`, `completions/{mean,min,max}_terminated_length`, `completions/clipped_ratio`, `reward_zero_std_groups`,

`completions/duplicate_frac`, and `sampling/importance_sampling_ratio/{mean,min,max}`.

— Names every metric but metrics are passive, not an action v1.18.0

THINNER COVERAGE BELOW

06 **HarnessAsyncGRPOTrainer for OpenEnv / Harbor agents**

Not checked yet NEW 55

Use `HarnessAsyncGRPOTrainer` in `trl.experimental.async_grpo.openenv_harness` for OpenEnv / Harbor agent training instead of constructing a `HarnessRolloutWorker` manually.

— Names the class and module but no usage example v1.18.0

07 **VLM checkpoint training on text-only data** Not checked yet NEW 45

Train VLM checkpoints (Qwen3.5, Qwen3.6, Qwen3-VL) on text-only data with the vision tower frozen via `AsyncGRPOTrainer`.

— Names the models but gives no further mechanism or example v1.18.0

08 **AMD ROCm and AWS Trainium backend support** Not checked yet NEW 45

TRL's trainers now run on AMD Instinct GPUs through PyTorch's ROCm build, and on AWS Trainium through TorchNeuron, PyTorch's native Neuron backend.

— States the backends but no setup detail or example v1.18.0

└──▷ **BREAKING ON UPGRADE**

! The `trl.experimental.async_grpo` submodule layout is gone: `from trl.experimental.async_grpo.async_grpo_config import AsyncGRPOConfig` and imports from `async_rollout_worker`, `packing`, `vllm_client`, or `weight_transfer` in that package no longer work. Import `AsyncGRPOTrainer` and `AsyncGRPOConfig` from `trl`, `AsyncRolloutLoop` and `Rollout` from `trl.trainer.async_grpo_trainer`, and `AsyncVLLMClient` (renamed from the old vLLM client) from `trl.generation.async_vllm_client`.

Our best guess

Check this claim Claim history

v1.18.0

! The `weight_transfer=` constructor argument on `AsyncGRPOTrainer` is removed; weight sync is now configured entirely through the config (NCCL or LoRA adapter sync).

Our best guess

Check this claim Claim history

v1.18.0

! The `rollout_worker=` constructor argument on `AsyncGRPOTrainer` is removed; to change rollout behavior, subclass `AsyncRolloutLoop` and pass it as `loop_cls`.

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.max_inflight_tasks` is renamed to `max_concurrent_rollouts` (fixed ceiling, default 512, no longer derived from `per_device_train_batch_size`).

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.heartbeat_stale_after_s` is renamed to `rollout_worker_timeout` (same 300 s default).

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.request_timeout` and `weight_sync_timeout` are removed; all vLLM timeouts are now covered by `vllm_server_timeout` (default 1800 s, was 240 s for readiness and 600 s for requests).

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.queue_maxsize` is removed; generation backpressure is now controlled by `max_staleness` in tokens.

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.fork_threshold_tokens` is removed; multi-turn rollouts always fork on re-tokenization drift.

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.token_budget <= 0` (fixed-count batching) is removed; the token-budget batcher is the only mode. Use `None` (reads `max_model_len` from the server) or a positive integer.

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.repetition_penalty` is removed because the sampler's log-probs include it while the trainer's do not, biasing the importance ratio.

Our best guess

Check this claim Claim history

v1.18.0

! `AsyncGRPOConfig.remove_unused_columns` now defaults to `False`; setting it to `True` raises an error because every dataset column is forwarded to the reward functions.

Our best guess

Check this claim Claim history

v1.18.0

! The Harbor example's `--max-inflight` flag is renamed to `--max-concurrent-rollouts`; the OpenEnv / Harbor environment server's `MAX_CONCURRENT_ENVS` must be at least `max_concurrent_rollouts + 1`.

Our best guess

Check this claim Claim history

v1.18.0

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

05 LM Studio

Introducing Agent Remote Control (2026-11-16)

Local LLM Runtimes

LM Studio is a desktop application that lets users run large language models locally on their computers without internet connectivity.

// WHY IT MADE THE LIST

DEPTH 4/5 IMPACT 4/5 NOVELTY 4/5

LM Studio's Agent Remote Control lets you manage Bionic agent sessions running on remote laptops, dev boxes, and cloud VMs from any device over LM Link, with a new headless `llmster` daemon for display-less machines and mid-session machine handoff — the first local LLM runtime to offer remote agent session management without a cloud intermediary.

> LM Studio's only release this window introduces Agent Remote Control, letting users manage Bionic agent sessions on remote machines from any device over an encrypted LM Link.

└──▷ **WHAT SHIPPED · 1 FEATURE** most completely described first ? what's the number?

01 Agent Remote Control for Bionic agent sessions

Not checked yet

NEW

91

LM Studio adds Agent Remote Control, letting you manage Bionic agent sessions on remote laptops, dev boxes, and cloud VMs from any device, encrypted end-to-end over LM Link (built with Tailscale). Enable remote hosting on a machine via the Remote control button at the top of the sidebar and toggling on Allow remote agent control; add and manage remote machines from the controlling device under `Settings > Devices > Remote Machines`. Run Bionic headless on display-less machines using `llmster`, a new headless daemon. Hand off work mid-session to another machine by typing `@` in the composer and picking a device — Bionic starts a session there and waits for it. You can also browse remote projects and chats from the sidebar under Remote machines, or choose which device a project runs on at creation time from a list of all machines in your device network with their online status.

.
.
.

Turn a headless cloud VM into a remote agent host so you can offload long-running tasks without keeping your laptop open.

📍 Install Bionic on the VM, then run: llmster On the VM (via SSH or remote session), enable remote control: click Remote control in the sidebar and turn on Allow remote agent control. On your laptop, go to Settings › Devices › Remote Machines and add the VM to your device network.

Hand off a long build-and-test job to a dedicated Windows machine mid-session while staying on your Mac.

> 📍 In the Bionic composer on your Mac, type @ and select your Windows machine. Bionic opens a new session there and waits for it, while showing you which session dispatched the work on the remote machine.

— Names exact UI paths, toggle, daemon, and handoff syntax.

Introducing Agent Remote Control (2026-…

└──▷ **ALSO FROM THESE RELEASES**

GOOD PICK? 👍 👎 GOOD REASON? 👍 👎

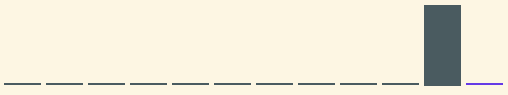
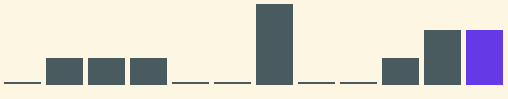
Where each pick stands · Nov 13–19

Tools with changes in each pick's category over the last 12 weeks, including the pick itself.

TensorRT-LLM Category unknown		–
ONNX Runtime Inference Serving & Optimization		25
Amazon Bedrock AgentCore ML Platforms & SDKs		108
TRL Model Training & Fine-Tuning		28
LM Studio Local LLM Runtimes		28

Each pick's release rhythm · Nov 13–19

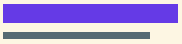
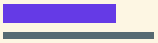
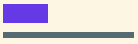
Changes per week for the last 12 weeks. The bright bar is this week; a dash marks a week with no daily.

TensorRT-LLM		44 in 12 known wk
ONNX Runtime		11 in 12 known wk

Amazon Bedrock AgentCore		37 in 12 known wk
TRL		10 in 12 known wk
LM Studio		12 in 12 known wk

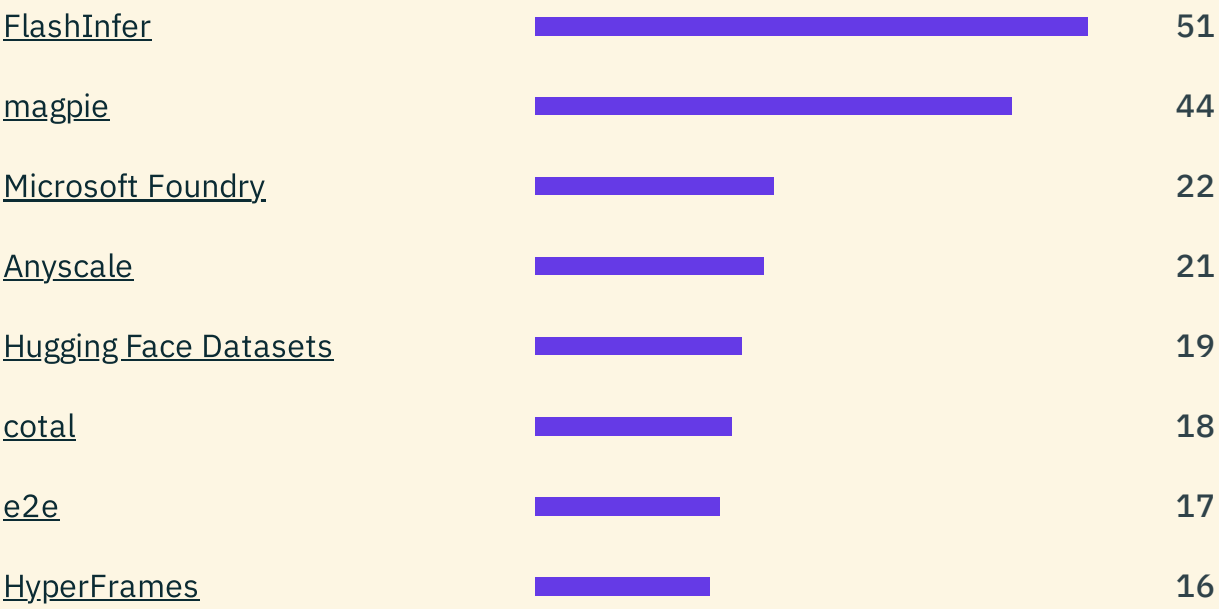
Which categories moved · Nov 13–19

The thick bar is this week. The thin bar is the average of the four weeks before, counting only weeks with a daily.

AI Coding Agents		211 usual 291.8
AI Agent Frameworks		126 usual 234
Inference Serving & Optimization		96 usual 80.5
Inference Providers & GPU Clouds		84 usual 52.2
ML Platforms & SDKs		62 usual 83.2
MCP Servers & Tooling		53 usual 120
Vector DBs & RAG		46 usual 105.5
AI Observability & Evals		24 usual 71.8
Model Training & Fine-Tuning		24 usual 40
Open-Weight Models		23 usual 56.2
AI App Builders		20 usual 1.8
Frontier Models		20 usual 38

Who shipped the most · Nov 13–19

Changes per tool this week. Volume is not a quality score.



Upcoming conferences →

// END OF TRANSMISSION

The AI Toolchain · This Week's Highlights · November 20, 2026
The full firehose — every feature release, every day — is in the newsletter.

▷ Subscribe