GOOGLE AI × COLAB · TESTED NOV 22–23, 2026

Your Google AI plan comes with Colab GPU time. Here's what it actually gets you.

I spent a night benchmarking the Colab compute units bundled with Google AI Pro: what each plan includes, what the GPUs really cost per hour, llama.cpp numbers on both A100s, and the setup that works.

AI Pro allowance	A100 40GB time	Llama 3.1 70B Q4_K_M	Black Friday annual Pro
200	**~37 h**	**24.6 tok/s**	**$0.23/h**
compute units a month	a month on Pro	A100 80GB, all on GPU	effective A100 40GB cost

TL;DR

- Since Sep 22, 2026, every paid Google AI plan includes Colab compute units: 50 a month on Plus, 200 on Pro, 1,000 or 2,000 on Ultra.
- On Pro I got both the A100 40GB and, with High-RAM on, the A100 80GB. A 70B model ran at about 24 tokens/sec on each.
- My whole night of testing (three benchmark runs, two builds, ~75 GB of model downloads) used about 8 units, 4% of a month on Pro.
- The GPU rates I measured on two cards were 13–18% higher than the published March figures. Check the Resources panel, not old tables.
- Annual Pro is $99.99 for the first year in the Black Friday sale, which works out to about $0.23 an A100 hour.

What each plan includes

Plan	Price (US list)	Units/month	What you get in Colab
Google AI Plus	$4.99/mo	50	Priority access to faster GPUs
Google AI Pro	$19.99/mo or $199.99/yr	200	Priority access, up to the A100 80GB (confirmed)
Google AI Ultra 5x	$99.99/mo	1,000	Adds H100 and RTX PRO 6000, background execution
Google AI Ultra 20x	$199.99/mo	2,000	Same as Ultra 5x

The fine print

- **Paid plans only.** Free and promotional trials and storage-only Google One plans get no Colab units.
- **Family plans:** the units go to the plan manager only. Colab requires an 18+ account.
- **Expiry:** each monthly batch expires 90 days after it's issued, so up to three months can build up (about 600 units on Pro).
- **Stacking:** units add to any standalone Colab Pro or Pro+ balance you already have.
- **Running out:** there's no overage bill. You drop back to free-tier rules, and running jobs can be cut off.
- **Background execution:** only Ultra keeps jobs running with the tab closed, for just under 24 hours. On Plus and Pro, an idle or closed tab eventually disconnects.

What the GPUs cost in units

GPU	VRAM	Units/hour	Where the rate comes from	Hours on Pro (200 units)
T4	16 GB	1.19	published	~168
L4	24 GB	1.94	measured · +13% vs 1.71	~103
A100 40GB	40 GB	5.40	measured · matches	~37
A100 80GB (High-RAM)	80 GB	8.88	measured · +18% vs 7.52	~23
RTX PRO 6000 (G4)	96 GB	8.71	published · Ultra only	–
H100	80 GB	~18	reported · Ultra only	–

Published rates are from Chris McCormick's Colab GPU table (Mar 2026). VRAM is marketed capacity. Your live rate is in the notebook's Resources panel (click the RAM/Disk indicator at the top right).

Benchmarks: llama.cpp on the A100s

Model	GPU	Size	Prompt, pp512 (tok/s)	Generation, tg128 (tok/s)
Llama 3.1 8B Q4_K_M	A100 40GB	4.58 GiB	4,891	154.4
Llama 3.1 70B IQ3_XS	A100 40GB	27.29 GiB	516	24.2
Llama 3.1 70B Q4_K_M	A100 80GB	39.59 GiB	592	24.6

- **High-RAM buys quality, not speed.** The 80GB card runs the better 70B Q4_K_M at the same speed the 40GB card manages with the heavily squeezed IQ3_XS.
- **Both cards hit about half their theoretical memory bandwidth** (~1.55 TB/s and ~2 TB/s) during generation, which is typical for llama.cpp.
- **Downloads are fast.** The 42.5 GB Q4_K_M came down from Hugging Face in 1 min 50 s (~385 MB/s).
- **The High-RAM machine is big:** 167 GB of system RAM, plus 236 GB of disk and 368 GB of scratch.

All runs used `-ngl 99` (every layer on the GPU), on Nov 22, 2026. The 40GB and 80GB runs used different llama.cpp commits (f4f6f2e and 44d5087).

What an A100 hour actually costs

Option	A100 40GB	A100 80GB
AI Pro, Black Friday annual ($99.99/yr = $8.33/mo)	**~$0.23/h**	**~$0.37/h**
AI Pro monthly or Colab Pro (~$0.10/unit)	~$0.54/h	~$0.89/h
RunPod Community Cloud	—	from $1.19/h
RunPod Secure Cloud	—	from $1.59/h

- **The deal:** up to 50% off annual Pro and AI Plus. Existing Google AI and Google One storage subscribers get early access Nov 20–26 (offers vary by account), and new subscribers can buy from Nov 27. Ultra isn't included, and annual Pro renews at $199.99 after year one.
- **The catch:** a rental gives you a guaranteed GPU, root access and storage that persists. Colab gives you priority, not a guarantee, sessions that end, and a fixed monthly allowance. At list price, every plan's units are worth about what the plan costs at the standalone ~$0.10/unit rate.

How to run it: the notebook route that works

1 · Pick the GPU

Open colab.research.google.com with the account that holds the plan. Go to Runtime → Change runtime type → A100 GPU, and turn High-RAM on for the 80GB card. Connect, then note your balance in the Resources panel. Check the card:

```
!nvidia-smi -L
```

2 · Build llama.cpp once and keep it in Drive

Any GPU runtime can do this, because the architecture flag decides what the build runs on. An L4 (~1.9 units/hour) built an A100-ready copy for me. CPU runtimes can't: they have no CUDA toolkit. The two flags cut the build from 12+ minutes to a fraction of that, by compiling for the A100 only and building just three programs.

```
from google.colab import drive
drive.mount('/content/drive')

!git clone --depth 1 https://github.com/ggml-org/llama.cpp /content/llama.cpp
!cd /content/llama.cpp && cmake -B build -DGGML_CUDA=ON -DBUILD_SHARED_LIBS=OFF -DCMAKE_CUDA_ARCHITECTURES=80 && cmake --build build -j --config
Release --target llama-bench llama-cli llama-server
!mkdir -p /content/drive/MyDrive/llama-bin && cp /content/llama.cpp/build/bin/llama-* /content/drive/MyDrive/llama-bin/
!/content/drive/MyDrive/llama-bin/llama-cli --version

drive.flush_and_unmount()   # let Drive finish uploading before you delete the runtime
```

3 · Every A100 session: copy the build back, download, benchmark

Don't keep models in Drive. Hugging Face is faster than copying them back out.

```
from google.colab import drive
drive.mount('/content/drive')
!cp -r /content/drive/MyDrive/llama-bin /content/ && chmod +x /content/llama-bin/*

!hf download bartowski/Meta-Llama-3.1-8B-Instruct-GGUF Meta-Llama-3.1-8B-Instruct-Q4_K_M.gguf --local-dir /content/models
!hf download bartowski/Meta-Llama-3.1-70B-Instruct-GGUF Meta-Llama-3.1-70B-Instruct-Q4_K_M.gguf --local-dir /content/models

!/content/llama-bin/llama-bench -m /content/models/Meta-Llama-3.1-8B-Instruct-Q4_K_M.gguf -ngl 99
!/content/llama-bin/llama-bench -m /content/models/Meta-Llama-3.1-70B-Instruct-Q4_K_M.gguf -ngl 99
```

4 · Actually talk to it

`llama-server` loads the model once and answers through an OpenAI-compatible API:

```
!nohup /content/llama-bin/llama-server -m /content/models/Meta-Llama-3.1-70B-Instruct-Q4_K_M.gguf -ngl 99 --port 8080 > /content/server.log 2>&1 &
!until curl -s localhost:8080/health | grep -q ok; do sleep 5; done
!curl -s localhost:8080/v1/chat/completions -H "Content-Type: application/json" -d '{"messages":[{"role":"user","content":"Explain PCIe 5.0 in two sentences."}]}'
```

5 · Shut it down

Runtime → Disconnect and delete runtime. Closing the tab isn't enough: an allocated runtime keeps billing, even when idle.

Hugging Face token

This is optional for public GGUF repos, but gated models need it, and anonymous downloads get rate-limited harder. Make a fine-grained read-only token. Add it under the key icon in the left sidebar, then turn on Notebook access.

```
import os
from google.colab import userdata
os.environ["HF_TOKEN"] = userdata.get("HF_read")
```

Secrets only work from the notebook UI. CLI runtimes can't read them.

The Colab CLI

- **Works from Windows, unofficially.** Google lists only Linux and macOS (WSL on Windows), but native PowerShell ran everything for me except one-line `ssh -- '...'` commands and `exec`, which were unreliable.
- **Signing in:** your first command (`colab sessions`) opens the browser sign-in. `colab auth` is something else: it gives the VM Google Cloud credentials.
- **The basics:** `colab new --gpu A100 --high-mem -s a100`, `colab ssh -s a100`, `colab stop -s a100`.
- **Separate sessions:** CLI runtimes and browser runtimes don't mix. `colab url -s a100 --open` opens a CLI runtime in the browser.
- **Known bug:** sessions can freeze after ~80 s or vanish between commands (issue #14). Build in the notebook.

Gotchas I hit, so you don't have to

Symptom	Cause	Fix
`session not found` when SSHing to a listed runtime	It was started in the browser (`[?]` in `colab sessions`)	Use it in the notebook, or `colab new` from the CLI
`Torch not compiled with CUDA`, no `nvidia-smi`	A bare `colab ssh` landed on a CPU runtime	Stop it, then SSH into the GPU session by name
Got 40GB instead of 80GB	"Standard" shape	Turn on High-RAM, or use `--high-mem`
llama.cpp build takes 12+ minutes	Compiles every GPU architecture and every tool	`-DCMAKE_CUDA_ARCHITECTURES=80` plus `--target`
`nvcc` not found	CPU runtimes have no CUDA toolkit	Build on a GPU runtime (an L4 is cheapest)
`huggingface-cli is deprecated`	The CLI was renamed	Use `hf download`
Units drain with nothing running	Idle runtimes still bill	Disconnect and delete the runtime; check `colab sessions`

Sources: Google Developers Blog, "Colab is now part of your Google AI plan" (Sep 22, 2026) and "Introducing the Google Colab CLI" · Colab FAQ · Google One Help, "Use Google AI Plus benefits" · googlecolab/google-colab-cli README and issue #14 · Chris McCormick, "Colab GPUs Features & Pricing" (Mar 2026) · Runpod GPU pricing · Beam, "NVIDIA A100 Pricing" · Vikas Kansal on X (Nov 20, 2026). All benchmarks, rates and balances are my own measurements on Google AI Pro, Nov 22–23, 2026.