imaPRO.in

PRACTICAL AI GUIDES



Kaggle TPU

A simple setup guide

Open a notebook. Complete verification.
Select the TPU. Check what your cloud session provides.

Our live setup: access accepted; hardware still queued.

No measured RAM amount or successful TPU calculation is claimed.

Simple English | Real screenshots | Prepared 28 November 2026

The old 330 GB figure is explained inside; it is not a promised allocation.

Start here

What you will get, and how to use this ebook

You can use a temporary cloud computer through Kaggle and request a TPU for suitable code. A TPU is a chip designed to help with some machine-learning calculations. It does not add RAM to your laptop.

Before you start

- **Have a Kaggle account**
  Sign in at kaggle.com in a computer browser.
- **Keep your phone nearby**
  You may need a code for phone verification and a separate identity check.
- **Expect availability to vary**
  An approved account can still wait in a queue for a machine.

What our walkthrough actually reached

On 28 November 2026, Kaggle accepted TPU v5e-8 after the user completed verification. The screen showed 20 hours remaining. The hardware check entered a queue, moving from #697 to #695. We saved the prepared notebook as version 2. No hardware output was produced.

Contents

2 Open a notebook
3 Complete the account checks
4 Select the TPU
5 Add the hardware check
6 Understand the queue and test
7 Save your work
8 Understand RAM, CPU and TPU
9 Fix common problems
10 Understand the code
11 Checklist and useful links

The cover is not numbered. The pages that follow use the printed page numbers shown in this contents list.

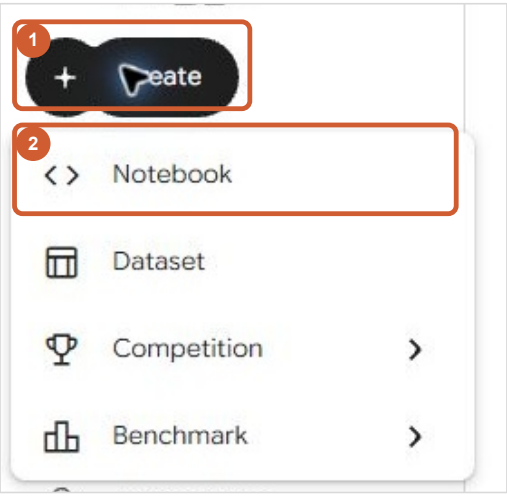
Open a notebook

Step 1 | Website: kaggle.com

A **notebook** is a page for code, notes and results. A **cell** is one box of code on that page.

1. **Open Kaggle**
   Go to kaggle.com and sign in.

2. **Create the notebook**
   Click **Create**, then **Notebook**. You should see the code editor.



1 Create 2 Notebook. Enlarged crop from our real browser capture.

3. **Name the notebook**
   Click the title at the top. We used **imaPRO TPU Setup Live Walkthrough**.

What you should see now

A notebook title at the top, a code box in the middle and notebook options on the right. The initial code is a sample from Kaggle. We replace it with our own small check later.

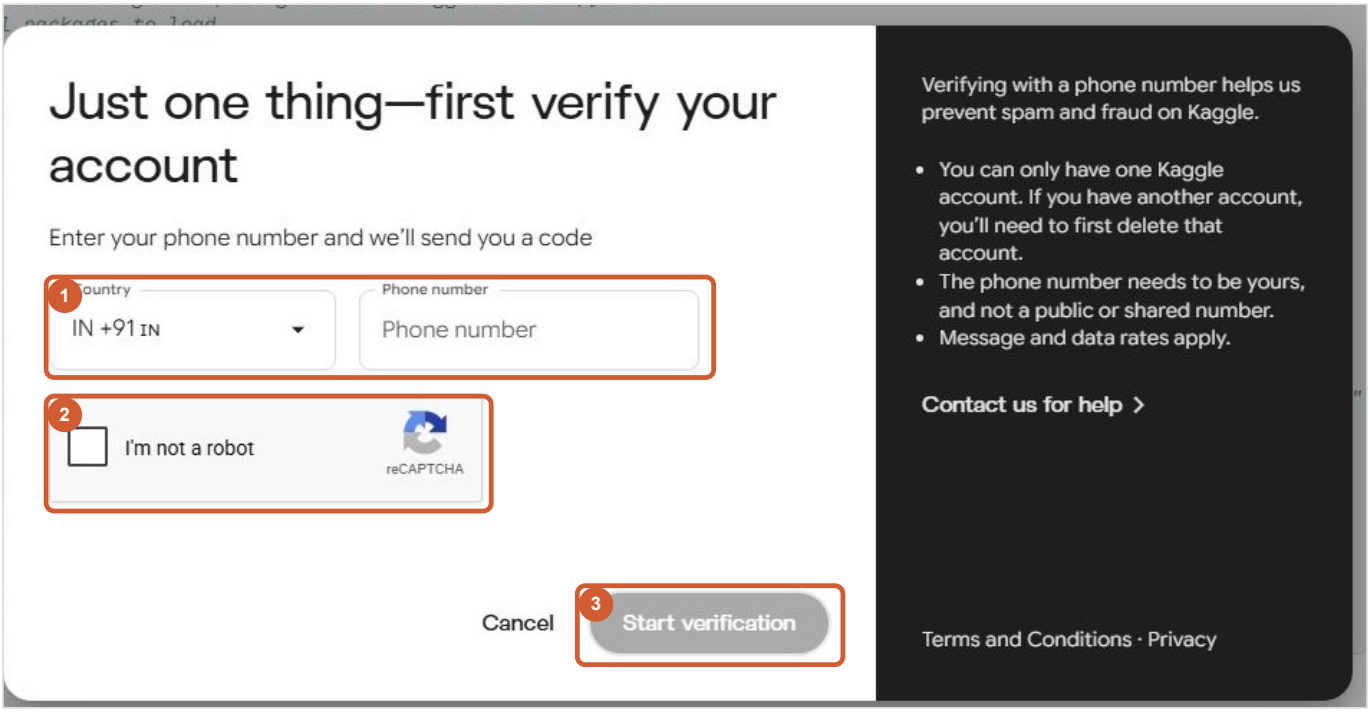
You do not need a Google Cloud billing account for this Kaggle notebook walkthrough. Opening a paid Cloud TPU VM directly is a different process.

Sources: S1 · S9

Complete the account checks

Step 2 | Phone verification and identity verification are separate

Open **Session options** in the right sidebar. If it shows **Get phone verified**, click that link.



1 Country and phone 2 CAPTCHA if shown 3 Request verification. The fields in this screenshot are empty.

1. **Verify the phone**
   Choose your country code and enter your number. Complete the CAPTCHA yourself if shown. Request the code and enter the code sent through Kaggle's verification flow.

2. **Complete the identity check if asked**
   When you select a TPU, Kaggle may open Persona. Follow its on-screen instructions yourself. It may ask for a document and/or a live face check.

3. **Return and try the TPU again**
   Completing a form is not always the final step. In our setup, the first return still showed the identity gate. TPU selection worked after the user finished the remaining face check.

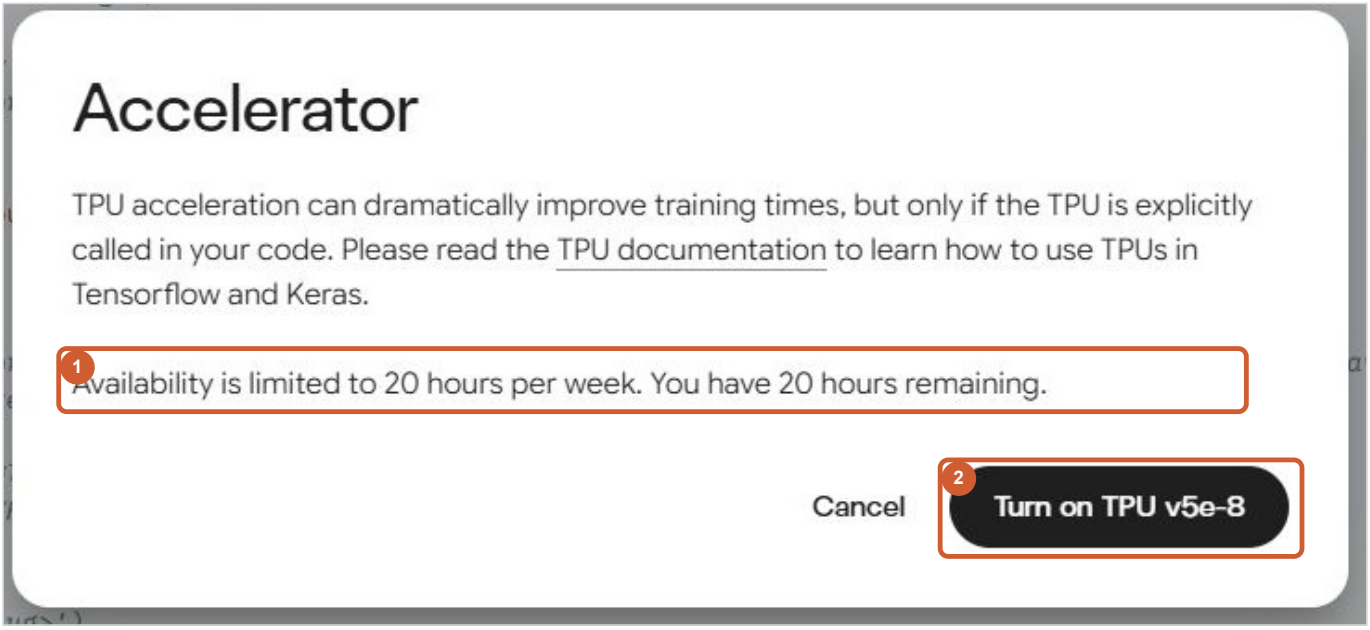
Keep private: your number, verification code, identity document and selfie. None belongs in a public screenshot or shared guide.

Sources: S3 · S4

Select the TPU

Step 3 | Settings > Accelerator > TPU v5e-8

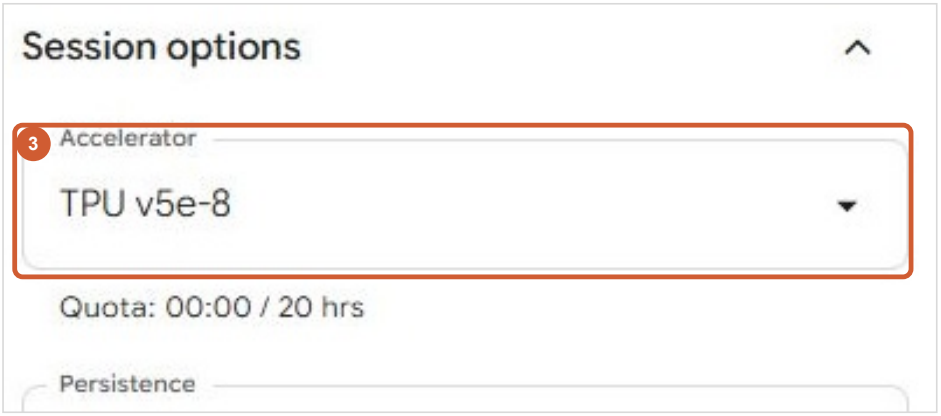
1 **Open the accelerator menu**
Click **Settings**, then **Accelerator**. Choose **TPU v5e-8**. An accelerator is an extra chip for supported calculations.



1 Our displayed allowance 2 The button that enables the selected TPU

2 **Read your allowance**
Our screen showed **20 hours remaining**. Read your own screen instead of assuming that a video's limits still apply.

3 **Enable the setting**
Click **Turn on TPU v5e-8**. The right sidebar should then show TPU v5e-8 as the selected accelerator.



3 TPU selected in Session options. A selected setting is not yet a running machine.

Sources: S2 · S5

Add the hardware check

Step 4 | Cell 1 reads the cloud computer you actually receive

A **runtime** or **session** is the temporary cloud computer running your notebook. The code below reads its system information.

Click inside the first code box. Select the sample text and replace it with this cell. Keep the spaces before indented lines.

```
import os
from pathlib import Path

mem = Path("/proc/meminfo").read_text()
total_kib = int(next(line.split()[1] for line in
    mem.splitlines() if line.startswith("MemTotal:")))
print("Host RAM in GiB:", round(total_kib / 1024**2, 1))
print("Logical CPUs:", os.cpu_count())

import jax
devices = jax.devices()
print("JAX version:", jax.__version__)
for device in devices:
    print(device.platform, device.device_kind)
tpus = [d for d in devices if d.platform == "tpu"]
if not tpus:
    raise RuntimeError("No TPU detected. Recheck the runtime.")
print("TPU devices detected:", len(tpus))
```



1 Run current cell 2 Our check entered in the notebook. Real browser capture.

Where this code came from

This check was written for the guide using Python's system tools and JAX's device-list function. **JAX** is a Python library for numerical work on supported chips. This code reads the assigned hardware; it does not unlock RAM.

What it should print after allocation

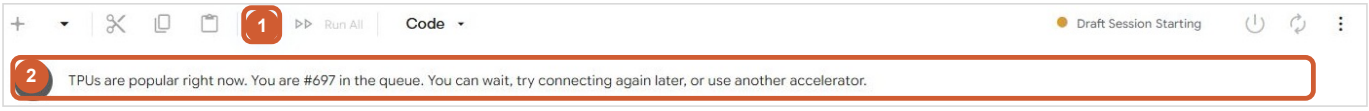
It should print host RAM, CPU count, JAX version and devices. Confirm that a TPU appears before running the TPU test.

Sources: S6

Run, then read the status

Step 5 | A queue is different from a successful calculation

Select the first cell and click the triangle-shaped **Run current cell** button. Kaggle requests a machine, then runs the cell when the session is ready.



1 Run 2 Our real queue message. It started at #697 and later moved to #695.

Our actual result: the first cell stayed queued. It printed no RAM or TPU result. Kaggle did not show a reliable start time. You can wait or return later.

Cell 2: run a tiny TPU calculation

Use this only after Cell 1 detects a TPU. It puts a small array on one TPU device and multiplies it. An array is a table of numbers.

```
import numpy as np
import jax

tpus = [d for d in jax.devices() if d.platform == "tpu"]
if not tpus:
    raise RuntimeError("Select a TPU runtime first.")

x = jax.device_put(np.ones((256, 256), np.float32), tpus[0])
y = jax.jit(lambda a: a @ a.T)(x)
y.block_until_ready()
print("Shape:", y.shape)
print("Check value:", float(y[0, 0]))
print("Devices used:", y.devices())
```

Expected output, not our observed result: shape (256, 256), check value 256.0, and a TPU in Devices used. Preparing the calculation for the first run can take time. This checks one device; it does not measure speed or test a large model.

Sources: S6 · S7

Save your work

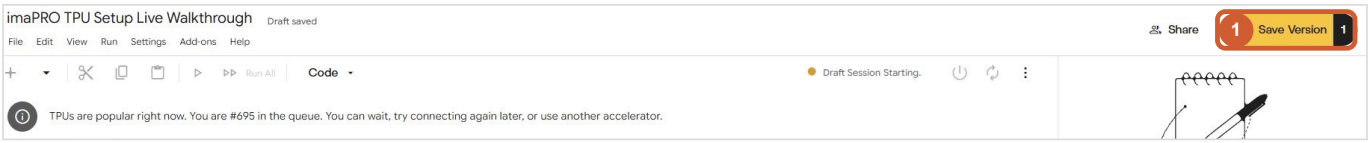
Step 6 | Keep the notebook and any results you need

Save the notebook

1. **Open Save Version**
   Click **Save Version** at the top right.

2. **Choose Quick Save**
   Quick Save keeps the notebook as it currently looks. It does not run cells that have not been executed.

3. **Name and save the version**
   We used **Setup prepared - TPU allocation queued**. The final saved notebook showed version 2.



1 Save Version. This capture shows the earlier counter; a later screen check confirmed final version 2.

Cell 3: save a result file after Cell 2 works

```
from pathlib import Path
Path("/kaggle/working/tpu-check.txt").write_text(
    f"Devices used: {y.devices()}\nShape: {y.shape}\n"
)
print("Saved: /kaggle/working/tpu-check.txt")
```

This file lives on Kaggle, not in your laptop's Downloads folder. Find it in **Output > /kaggle/working**. Refresh the folder list and use its available download control to keep a local copy.

Save/download files you need before the session ends. When your real work is finished, use the session or power control to stop it and avoid using more quota.

Our status: the notebook was saved. The result-file cell was prepared but not run, because the TPU was still queued.

Sources: S1

RAM, CPU and TPU explained

The old 330 GB headline refers to host memory

The **host** is the main cloud computer running Python. It works alongside the accelerator. These resources do different jobs.

Term	Simple meaning
Host RAM	Working memory for the main cloud computer. It is not extra memory installed in your laptop.
CPU	The cloud computer's general-purpose processor. The printed logical CPU count is not a count of TPU chips.
TPU	An accelerator for supported numerical and machine-learning code. Ordinary code does not automatically become faster just because a TPU is selected.
TPU device memory	Memory attached to the accelerator. It is different from the host RAM shown by the system file.
GiB	The memory unit printed by our check. It uses powers of 1024, so do not expect it to match a GB headline exactly.
Quota	Your allowed time or usage amount. Read the current allowance in your account.

What changed about 330 GB?

Kaggle's older notebook documentation listed a host with 96 CPU cores and 330 GB RAM. The old TPU v3 hardware was later retired. The current menu in our walkthrough offered TPU v5e-8.

There is no step in this guide that guarantees the retired host specification. The useful number is the RAM your own allocated session prints.

Kaggle and Google Cloud billing

This walkthrough stays inside Kaggle notebooks. Creating a Cloud TPU VM directly in Google Cloud is a separate, billed service. No paid Google Cloud subscription was started here.

Sources: S1 · S2 · S8 · S9

Fix common problems

Start with the account, selected hardware and session status

Problem	What to do next
TPU is gray or missing	Complete the phone and identity checks. Return to the notebook, refresh if needed, and review quota and availability messages.
Verification still appears	Finish any remaining step in the identity flow. A clickable menu entry alone did not prove approval in our setup.
TPU is selected but code waits	Read the queue or startup message. Wait or try later. A selected setting does not reserve a machine.
Only CPU is listed	Confirm that the requested TPU session actually started. Recheck the accelerator and rerun the first cell.
RAM differs from the video	Use the measured amount in your session. The old 330 GB example is not a current guarantee.
Python shows a package error	Use the current Kaggle environment first. Avoid randomly upgrading every library. Consult the official JAX instructions for that runtime.
A model does not fit	Start smaller. A large host RAM amount does not mean all accelerator memory is pooled for one model.
Quota is used up	Save your work and follow the allowance/reset information in your account. Availability is not unlimited.

When to ask for help

For account or identity problems, use Kaggle support. For JAX package problems, use the official installation guide. Describe the error without sharing private verification details.

Understand the code

What each block does, in everyday language

Cell 1: read and list

Code	Purpose
Path("/proc/meminfo").read_text()	Reads a Linux system file containing memory information.
MemTotal:	Selects the line reporting total host memory.
total_kib / 1024**2	Converts the reported KiB amount to GiB.
os.cpu_count()	Reads the cloud host's logical CPU count.
jax.devices()	Lists the computation devices JAX can see.
if not tpus	Stops with a clear message when no TPU is detected.

Cell 2: put and calculate

np.ones builds a small table filled with ones. **jax.device_put** places it on the selected TPU. **jax.jit** prepares the calculation for the device. **block_until_ready** waits for the result before printing.

The result is a square table with 256 rows and 256 columns. Because each dot product adds 256 ones, the example check value is 256.0. This is the expected arithmetic, not a result from our queued notebook.

Cell 3: write a small text file

Path(...).write_text creates the text file in /kaggle/working. It records the device and result shape from Cell 2, so run it only after Cell 2 succeeds.

About brackets and spaces

Copy the code exactly. Round brackets contain function inputs. Square brackets select items or build lists. Curly braces inside the final f-string insert values. Keep the indented lines under **for** and **if**; Python uses those spaces to group the code.

Sources: S6 · S7

Checklist and useful links

Keep this page beside your notebook

Before and after a run

1. **Check**
   Both required account checks are complete.
2. **Check**
   TPU is selected and the session has actually started.
3. **Check**
   Cell 1 prints the actual host RAM and lists TPU devices.
4. **Check**
   Cell 2 prints the expected shape and value on a TPU.
5. **Check**
   Files you need are saved or downloaded.
6. **Check**
   The current quota is understood and the finished session is stopped.

Official reading

Kaggle notebook documentation

Kaggle staff announcement on TPU v5e

Kaggle identity verification announcement

Kaggle staff on phone and identity checks

Kaggle TPU documentation

JAX device listing reference

JAX array placement reference

Google Cloud TPU v5e hardware

Google Cloud TPU pricing

More imaPRO guides

Claude animation styles and real examples
Make a browser game with Gemini

These are other creative coding guides. Their workflows are not claimed to require a Kaggle TPU.