HACKBLOX 2026 · AI TRACK

3LC × HACKBLOX Scene Classification Challenge

Use 3LC to improve your data, train a classifier, and climb the leaderboard

Overview

Can you tell a glacier from a mountain? (Harder than it sounds — this dataset genuinely confuses them.) Build a 6-class natural-scene classifier using data-centric AI with 3LC and compete on a Kaggle leaderboard. Your score is live as soon as you submit.

This challenge is the 3LC track under the AI track of HackBlox 2026: a 24-hour AI, Web3 and open-source hackathon by Hackers Cult. It runs across the Round 2 window: **Sat 5 Sep, 10:00 AM IST** → **Sun 6 Sep, 10:00 AM IST**.

You'll train on a few labeled samples, use 3LC to collect metrics and predictions on unlabeled data, label a batch using the 3LC Dashboard with model feedback (embeddings), retrain, watch your score improve, and repeat until you get the best score.

What is data-centric AI?

Traditional ML focuses on improving the model. Data-centric AI focuses on improving the data.

"Data-centric AI is the discipline of systematically engineering the data used to build AI systems." — Andrew Ng

In this competition you will:

- Start with 600 labeled images (100 per class) and 6,000 unlabeled images
- Classify scenes into 6 classes: buildings (0), forest (1), glacier (2), mountain (3), sea (4), street (5)
- Use the 3LC Dashboard to visualize embeddings and analyze model behavior
- Strategically label unlabeled samples using model feedback and embeddings — your final train table may have at most 3,000 rows with weight > 0 (the 600 seed labels count toward this), so choose wisely
- Improve accuracy by curating data, not by changing the model (ResNet-18 is fixed)
- Train from scratch — no pretrained weights

What makes this competition interesting?

Traditional competition	This competition
Optimize model architecture	Fixed model (ResNet-18)
Use all data as-is	Strategic labeling of unlabeled pool
Model-centric	Data-centric

The goal is to get the best test accuracy by improving your dataset with 3LC.

Scoring

- **Metric:** Classification accuracy on the hidden test set (0–1, higher is better)

- **Leaderboard:** Live; each submission is scored against the solution file
- **Public / Private:** 50% of test used for public leaderboard (900 images), 50% for final (private) ranking (900 images), stratified by class
- Winners are determined by the private score and offline evaluation

Required tool: 3LC

This competition uses the 3LC data-centric AI platform:

- **Tables** — versioned datasets; every edit creates a revision
- **Runs** — experiment tracking with per-sample metrics
- **Sample weights** — control which samples are active (weight = 1) or inactive (weight = 0)
- **Embeddings** — visualize how the model clusters images in the Dashboard (3D by default), colored by confidence or by predicted label

Get Started

- 3LC Documentation
- Video walkthrough (start here — made for this challenge): Part 1 – Setup, First Training Run & Kaggle Submission; Part 2 – Improving Your Score with Embeddings-Guided Labeling
- Deeper 3LC reference from a previous challenge: Parts 1–3

Final Submission & Evaluation Form (Required)

In addition to submitting predictions on Kaggle, you must complete the evaluation form by the competition deadline to be eligible for prizes and certificates.

Form: https://docs.google.com/forms/d/e/1FAIpQLSeWCQh6vcc5oigeyDluOf05Nfk5w03dgchasiFVH_TyRos9UQ/viewform

When to submit: Before Sun Sep 6, 10:00 AM (IST)

What we ask for: Your team name (as on Kaggle and on Unstop), team member names and emails, your private leaderboard rank and test accuracy, and a link to your private GitHub repository with your code, write-up, and 3LC tables/screenshots. You will also be asked to add the judge (Rishikesh-Jadhav) as a collaborator on that repo so it can be verified.

Your GitHub repo should contain:

- Code (.py files)
- Write-up / presentation (PDF or MD)
- Zipped 3LC project — Windows: *C:\Users\USERNAME\AppData\Local\3LC\3LC\projects\Intel-Scene*; macOS/Linux: *~/.local/share/3LC/projects/Intel-Scene*
- 3LC dashboard screenshots (embeddings / accuracy)
- README

Why it matters: this is used to verify results, award prizes and certificates, review your data-centric approach (embeddings, labeling, weighting), and send certificates and follow-up. Your feedback also helps improve future hackathons. After winners are announced, you're free to make your repo public and share it on LinkedIn to showcase your data-centric AI approach.

Please complete the form even if you don't place in the top ranks — it's used for participation certificates and to learn from your experience.

Timeline

Milestone	Date / Time (IST)
Competition start	Sat Sep 5, 10:00 AM
Final submission	Sun Sep 6, 10:00 AM
Winners announced	After offline evaluation is complete

Rules (Summary)

- **Model:** ResNet-18 only — no other architectures
- **Training data:** Only data provided (train/val); no external image data
- **Pretrained weights:** Not allowed — train from scratch
- **3LC:** Required for the data-centric workflow
- **Labeling budget:** Final train table max 3,000 weight-1 rows (600 seed labels included); verified from 3LC table lineage during offline evaluation
- **Submissions:** Follow Kaggle submission limits for this competition
- **Team size:** Maximum 4 members per team; your Kaggle team must match the team registered for HackBlox

Prize eligibility: Every team member needs a 3LC account, the Team Captain must fill out the Evaluation and Feedback Form (one per team) before the deadline, and teams must abide by the rules during the hackathon. Any form of cheating results in disqualification.

Support & Community

- 3LC questions (starter kit, tables, Dashboard, embeddings): join the 3LC Discord, #hackathons channel
- HackBlox questions (teams, tracks, general hackathon): HackBlox Discord or support@hackerscult.online

Acknowledgments

- 3LC — for powering the challenge with the data-centric AI platform and support
- Hackers Cult — for organizing HackBlox 2026 and hosting this challenge under the AI track

Good luck, and may the best data win.

Description

Goal

Build a 6-class scene classifier (buildings = 0, forest = 1, glacier = 2, mountain = 3, sea = 4, street = 5). You compete on test set accuracy; the model architecture (ResNet-18) is fixed. The twist: you improve your score by improving your data with 3LC, not by changing the model.

Data

- **Train:** 600 labeled (100 per class) + 6,000 unlabeled (undefined). Use 3LC to label part of the unlabeled set and curate which samples to train on. Final train table may have at most 3,000 rows with weight = 1 (seed labels included)
- **Val:** 1,200 images (200 per class, balanced); use for monitoring during development — gives stable feedback so the model generalizes well
- **Test:** 1,800 images, hidden labels; you only get images. Predictions go into submission.csv and are scored on Kaggle

Data-Centric Workflow

- **Train** — run the starter train.py (or your own pipeline). Training creates a 3LC Run with per-sample metrics and embeddings
- **Analyze** — run the 3lc service and open the 3LC Dashboard. Inspect embeddings, confusion, and hard examples
- **Fix data** — in 3LC: correct wrong labels, label useful undefined samples, set sample weights. Save a new table revision
- **Retrain** — run training again; it uses the latest table revision (e.g. .latest())
- **Submit** — run predict.py to get predictions on the test set, create submission.csv, and upload to Kaggle

Repeat the cycle to improve test accuracy.

Classes

Label	Class
0	buildings
1	forest
2	glacier
3	mountain
4	sea
5	street

What You Will Learn

- Data-centric AI — improving models by improving data
- Active learning — choosing which unlabeled samples to label

- 3LC — tables, runs, embeddings, and sample weights
- Experiment tracking — comparing runs and table revisions

Resources

- Videos: Part 1 – Setup & First Submission · Part 2 – Embeddings-Guided Labeling
- Starter kit: register_tables.py, train.py, predict.py, config.yaml, README.md, sample_submission.csv
- 3LC Docs · 3LC GitHub

Planning Your 24 Hours

- Setup and first training run take about 20 minutes
- Budget the bulk of your time for label → retrain loops — two or three good loops beat one large indiscriminate labeling batch
- Submit an early baseline so you always have a score on the board

Evaluation

Metric

- Accuracy on the hidden test set
- Score is between 0 and 1 (higher is better)
- Predictions are compared to the ground truth labels in the solution file

Submission Format

Your submission must be a CSV with exactly these columns: *image_id, prediction, confidence*. Example rows:

image_id	prediction	confidence
`test_00001`	`3`	`0.92`
`test_00002`	`0`	`0.88`
`...`	`...`	`...`

Column	Type	Description
image_id	string	Unique identifier; must match the IDs in sample_submission.csv
prediction	int	Class label: 0 = buildings, 1 = forest, 2 = glacier, 3 = mountain, 4 = sea, 5 = street
confidence	float	Model confidence for the predicted class (0.0–1.0)

- **image_id:** must match the test set IDs (test_00001 … test_01800). One row per test image
- **prediction:** integer 0–5 only
- **confidence:** typically max(softmax(logits)). Not used in the metric — if missing entirely, auto-filled with 0.5; if present, values must be numeric and in [0.0, 1.0]

Validation Rules

Submissions are rejected if:

- Missing column: image_id or prediction
- prediction is not an integer 0–5 (or not numeric), or any prediction is missing
- Any image_id from sample_submission.csv is missing from your file (extra image_ids not in the solution are ignored, not rejected)
- confidence is present but outside [0.0, 1.0], non-numeric, or missing values

Notes on lenient cases (not rejections):

- Duplicate image_ids are de-duplicated, keeping the first occurrence — only your first row per ID counts
- A missing confidence column is auto-filled with 0.5 and the submission scores normally

Use sample_submission.csv (included in the starter kit) as the template — it lists all 1,800 required image_ids and the column names.

Leaderboard

- **Public leaderboard:** 50% of the test set (900 images) — the score you see when you submit during the competition
- **Private leaderboard:** the other 50% (900 images); used for final ranking when the competition ends — you don't see this score until the end

The split is stratified by class (150 images per class in each half) so both sets share the same distribution — this prevents overfitting to the public leaderboard and ensures the final ranking reflects true generalization.

Make sure your submission covers the full test set (all 1,800 image_ids in sample_submission.csv). You submit one CSV; Kaggle computes both public and private scores from it.

Selecting Final Submissions

- You may mark up to 2 submissions as final for private-leaderboard judging (Submissions tab → “Use for Final Score”)
- If you don't select any, Kaggle defaults to your best-scoring submissions on the public leaderboard — which isn't always your best private score
- Do this before the deadline