YUCLAW

User Guide

v5.0 series · July 2026 · github.com/YuClawLab/yuclaw-brain

`Open-source, evidence-first financial research · MIT license · local inference, zero cloud LLM calls · git-anchored Verified Research Ledger`

This guide takes you from `pip install` to reading signals, tracing evidence to SEC filings, reproducing the published record on your own machine, and using the dashboards, exports, and programmatic surfaces. It assumes basic command-line familiarity and nothing else.

Read this first. YUCLAW is research and educational software. It is **not** financial advice, investment advice, or a recommendation to buy, sell, or hold any security. Signal labels are research classifications, not trade directions — there is no SELL or SHORT label anywhere in the system. Outputs are generated by automated models and may contain errors. The full disclaimer is in Section 10 and DISCLAIMER.md.

CONTENTS

1 What YUCLAW is (and is not)

YUCLAW ingests SEC filings (Form 4, 8-K, 10-Q, 10-K, 6-K, 40-F), extracts typed evidence events with a locally hosted LLM, checks every extraction against the source text with a deterministic verifier, and combines the evidence with market context into a nine-component composite research signal. Every day's signal set is content-hashed and committed to a public git repository (`yuclaw-trust`) before pages publish — so the record is tamper-evident and independently checkable.

Three design commitments define the project:

- **Every signal traces to a filing.** Each composite decomposes into its components, and every evidence event links to the SEC document it came from.
- **Every snapshot is hashed to a public ledger.** Git-anchored, never edited. Outages are disclosed, never backfilled.
- **Every published chart is reproducible.** `yuclaw replay-lab` rebuilds the cohorts and recomputes every statistic and ledger hash from published derived data, and exits non-zero on any mismatch.

What YUCLAW is not: it is not a trading system, does not connect to any broker, does not manage money, and does not produce buy/sell recommendations. The ingestion and extraction pipeline runs on the project's own hardware (a single NVIDIA DGX-class node); as a user you consume the published, hash-anchored outputs through the CLI, SDK, REST API, MCP server, and dashboards.

2 Install and your first signal

Requirements: Python 3.10 or newer, `pip`, and an internet connection. No GPU is needed to use published signals or reproduce the Validation Lab.

```
$ pip install yuclaw
$ yuclaw demo                        # 3-minute guided journey — works offline, zero config
$ yuclaw why AMD --as-of 2026-05-20 # bundled offline signal, no backend needed
```

Live signals for **all** tickers need the local backend (`docs/v4/backend_setup.md`); the published-data commands — `replay-lab`, `verify`, `validation` — work anywhere with no backend. With the backend running, `yuclaw why NVDA` produces output shaped like this (backend-mode output, abbreviated):

```
NVDA composite score: +0.299  (signal label: NEUTRAL)

Components (score × weight × confidence):
  C1 Momentum        +0.46   (weight 0.12)
  C4 Macro           +0.60   (weight 0.15)
  C6 Event Impact    +0.16   (weight 0.18)
  ... (all nine components listed)

Top contributing events (last 7 days):
  ↑  +0.02  2026-05-14  M_AND_A_CLOSE (d1 cascade)
             CASCADE d1 via HPE→NVDA (supply, w=0.15) from HPE: H3C divestiture
             source: https://www.sec.gov/Archives/edgar/data/1645590/...

Compliance: Research only. Not financial advice. Not a registered investment advisor.
```

Three things to notice: the label (`NEUTRAL`) is a research classification of the composite score; each component's contribution is shown with its weight; and every contributing event carries a link to the SEC document it was extracted from. If you take one habit from this guide, take this one: **follow the source links.** The engine is built to show its work — make it.

> **Version note (5.0.x).** The pip-installed 5.0.x releases predate the `events`, `lens`, `export`, and `memo` subcommands — they ship in v5.1. To use them today, run from a checkout of `main`: `git clone https://github.com/YuClawLab/yuclaw-brain` and invoke `python3 -m v3.cli ...`. Everything else in this guide works from the pip install.

3 Reading a signal

The nine components

The composite is a confidence-weighted sum of nine components. C6 (event impact) carries the highest single weight by design: evidence is meant to correct price-only signals, not echo them.

#	COMPONENT	WEIGHT	#	COMPONENT	WEIGHT
C1	Momentum	0.12	C6	Event impact (evidence)	**0.18**
C2	Volume *	0.08 *	C7	Peer correlation	0.10
C3	Sector velocity	0.12	C8	Supply-chain cascade	0.12
C4	Macro regime	0.15	C9	Model trust	0.08
C5	Oil / rates / FX	0.05			

* C2 (volume confirm) currently contributes no signal — confidence-gated to zero since v3.0 pending volume-feed wiring; live composite weights effectively renormalize across the remaining eight components. Repair-vs-deprecation decision scheduled for v5.2.

Signal labels

The public vocabulary is fixed: `STRONG_BULLISH · BULLISH · NEUTRAL · WATCH · WEAKENING · NEGATIVE_EVENT · BEARISH_WATCH · RISK_ALERT`. These bucket the composite score and risk state; exact thresholds are documented in `docs/methodology/backfill.md`. Two facts worth knowing: extreme labels (`STRONG_BULLISH`, `BEARISH_WATCH`) are rare by construction

— they require broad component agreement plus at least one material non-insider event — and there is deliberately no SELL or SHORT label: these are research classifications, not trade directions.

The universe

Signals cover a 79-name scoring universe (equities, sector ETFs, broad ETFs, macro instruments). Separately, a 49-filer **Canada Resources evidence tier** (XEG / ZEO / GDX / URNM constituents) is ingested and dashboarded but never scored — the boundary is machine-enforced and regression-tested. See Section 6.

4 Command reference

COMMAND	WHAT IT DOES
`yuclaw why TICKER`	Composite signal + ranked evidence with SEC source URLs.
`yuclaw replay TICKER --date DATE`	Point-in-time signal as of the end of DATE (leak-audited; uses only data available then).
`yuclaw replay-lab`	Reproduce the entire Validation Lab from the public bundle; exit 0 = exact match.
`yuclaw validation`	In-Sample Event Validation panels + forward tracking ledger.
`yuclaw verify TICKER --date DATE`	Verified Research Ledger integrity check: confirms a signal hasn't been edited since publication. Checks record integrity and timing — not investment merit.
`yuclaw events --ticker SU --since DATE`	Accepted-events export (derived data only).*
`yuclaw lens canada --lens XEG`	Canada lens summary-card data as JSON — the same numbers the page renders.*
`yuclaw export --lens GDX --format csv`	Lens events export; `--page` builds the evidence packet.*
`yuclaw memo --ticker SU --days 30`	Evidence memo — grounded, citation-checked, linted. Requires a full local deployment (Section 7).*
`yuclaw brief`	Personalized digest (uses ~/.yuclaw/profile.json).
`yuclaw watch add TICKER`	Manage your local watchlist.
`yuclaw profile show`	Show local preferences.

* Ships in v5.1; run from a `main` checkout meanwhile (see the version note in Section 2). Worked examples with real output: `docs/usage.md`.

5 Verify everything yourself

YUCLAW's central claim is not that its signals are good — that is for the data to prove or disprove in public. The claim is that **nothing is unverifiable**. Three tools back it:

5.1 Reproduce the Validation Lab

```
$ yuclaw replay-lab
# fetches the public replay bundle, rebuilds cohorts, recomputes every statistic,
# re-derives every ledger hash root — exit 0 on exact match, non-zero on ANY mismatch
```

This runs from a brand-new environment with nothing but the pip install. At the v5.0.0 release it reproduced every published daily ledger root and statistic exactly from a fresh venv. A standalone stdlib script (`tools/replay_lab.py`) does the same with no dependencies at all.

5.2 Check a specific signal against the public ledger

```
$ yuclaw verify AMD --date 2026-05-20 # works from bare pip — bundled ledger record
```

Recomputes the signal's hash and checks it against the git-anchored ledger in `yuclaw-trust` — independent confirmation that the record hasn't been edited since publication. For any other ticker/date, clone `github.com/YuClawLab/yuclaw-trust` next to your working directory (the full public ledger) or connect the local backend.

5.3 Replay any signal as of a past date

```
$ yuclaw replay NVDA --date 2026-06-02
```

Point-in-time filtering (`available_as_of ≤ as_of`) guarantees the replay uses only information that existed on that date. The same capability is exposed over REST (`/replay`) and MCP (`yuclaw_replay`).

5.4 Publish your replication

If you run a reproduction, the project wants your result — pass or fail. Open a GitHub issue using the replication template (it asks for OS, Python version, command, output hash, and result). Accepted reports appear on the public **Independent Replication Log**. A failed reproduction is a bug report of the highest value: the verifier exiting non-zero is exactly the alarm it exists to raise.

6 Dashboards and daily surfaces

SURFACE	URL / HANDLE	WHAT'S ON IT
Live Dashboard	`yuclawlab.github.io/yuclaw-brain`	Current signal set, regenerated daily after U.S. close.
Validation Lab	`.../validation_lab.html`	Decile-cohort event study; bootstrap CIs, Newey–West ICs, power meter; maturity gates printed on-page.
Canada Resources Evidence	`.../canada_resources.html`	Evidence dashboards for XEG / ZEO / GDX / URNM filers; 6-K/40-F prose path.
Open Index Evidence	`.../etf_evidence.html`	Index/ETF-level evidence views.
Today's Evidence Digest	`.../todays_evidence.html`	Daily change log: new filings, accepted events, posture changes, ledger root, replay status.
Independent Replication Log	`.../replication.html`	External reproductions of the record (honestly empty until someone submits one).
Telegram digest	`@yuclaw_signals`	Daily signal digest, 07:35 MT.

> **Reading the Canada Resources page.** The Canada tier is evidence-only: its names are ingested and dashboarded but never enter the scoring universe or the Lab's decile study. One disclosure worth understanding: most Canadian cross-listed (MJDS) filers report insider trades on Canada's SEDI system, not SEC Form 4 — so insider-derived statistics are marked *outside evidence scope* for those names rather than shown as misleading zeros. Coverage percentages and grades are printed per lens; thin coverage is excluded rather than padded.

Every page carries a freshness stamp. If the pipeline has an outage, the gap is disclosed on the page and never backfilled — a visible gap is the honest state of the record.

7 Evidence memos and data exports

Evidence memos

`yuclaw memo --ticker SU --days 30` produces a short research memo in which **every sentence cites an event ID** that resolves to a filing excerpt. Generation uses the local LLM; verification does not: a deterministic verifier checks each citation against source text, and a linter enforces a restricted conclusion vocabulary plus the research-only footer. Memos require a full local deployment (events database + local models) — pip-only users can read the approved demo at `docs/examples/evidence_memo_su.md` to see the exact output shape.

Evidence packets and exports

`yuclaw export --lens GDX --page` builds an evidence packet: a zip of typed events, verified excerpts, and replay data, each carrying citation metadata (`data_through, build_date, source_commit, ledger_root, methodology_version, scope_note, known_limitations`) so anything you quote downstream stays traceable.

> **Export rule.** Exports contain **derived YUCLAW data only** — typed events extracted from public SEC filings, scores, hashes, and the track record. Raw third-party market data (vendor price and options feeds) is never redistributed, per those vendors' terms. Citation format for research use: "YUCLAW v5.0.0, data through <date>, commit <hash>" — the packet metadata gives you the exact string.

8 Programmatic access: SDK, REST, MCP

- **Python SDK** — `pip install yuclaw`; the same signal, replay, and verify surfaces as the CLI, importable.
- **REST API** — FastAPI server (`v3/api/`) including the `/replay` endpoint for point-in-time queries.
- **MCP server** — FastMCP stdio server exposing 7 tools (including `yuclaw_replay`), for use from MCP-capable AI clients.
- **OpenClaw skill** — install via `yuclaw/openclaw/install.sh`, or run `python3 yuclaw/openclaw/mcp_server.py` (port 8002).

All inference in the pipeline is local to the project's node (Llama 3.1 70B + Gemma via one Ollama daemon) — there are no cloud LLM calls anywhere, and SEC EDGAR is the evidence layer's only external data source.

9 Honest limitations

These are stated up front on the site and repeated here because a user guide that hides them would defeat the project's point. Full detail: `docs/methodology/backfill.md`.

- **The forward record is young.** Look-ahead-free forward tracking began 2026-05-20. In the Lab's own words: "No forward alpha has been statistically proven yet." The on-page power meter quantifies what the current sample can and cannot detect.
- **In-sample results are optimistic by construction.** The In-Sample Event Validation panel was reconstructed after the fact by the replay engine, and the extraction model's training window overlaps it — treat in-sample numbers as an upper bound, not a promise.
- **C6 risk channel is partially confirmed.** C6 risk channel: rare-by-construction confirmed OOS (22% fire rate, n=9 held-out); sign positive at n=2 elevated — accruing. Deeper architecture layers are gated on that confirmation.
- **C4 macro regime is temporarily frozen** (as of 2026-05-18, with an on-page staleness disclosure) pending macro-engine restoration.
- **Small samples are labeled.** Panels with too few observations for inference say so on the page; no headline return tables appear without their n.

10 Troubleshooting, support, and disclaimer

SYMPTOM	CAUSE & FIX
`events / lens / export / memo` not recognized	Pip 5.0.x predates these subcommands. Run from a `main` checkout: `python3 -m v3.cli ...` (Section 2). They ship in v5.1.
`why TICKER` says it needs a backend	Expected from a bare pip install. Run `yuclaw demo` or `yuclaw why AMD --as-of 2026-05-20` (bundled offline signal); the published-record commands (`replay-lab`, `verify`, `validation`) work anywhere. Live signals for all tickers: `docs/v4/backend_setup.md`.
`replay-lab` exits non-zero	That is the alarm working. Re-run once (transient network), then open a GitHub issue with the replication template — a reproducible mismatch is a high-value report.
Dashboard looks stale	Check the freshness stamp on the page. Pages regenerate daily after U.S. market close; outages are disclosed on the Lab page, never backfilled.
`memo` fails without a database	Memos need the full local deployment (Section 7). Read the demo memo instead, or run the rest of the CLI, which works off published data.
Questions, bugs, ideas	GitHub issues on `YuClawLab/yuclaw-brain` · X: @Vincenzhang2026.

⚠ **Disclaimer.** YUCLAW is open-source research and educational software. It is NOT financial advice, investment advice, or a recommendation to buy, sell, or hold any security. All signals, scores, and analyses are generated by automated AI models and may contain errors. Past performance does not guarantee future results. Trading involves substantial risk of loss. You are solely responsible for your own investment decisions; consult a licensed financial advisor before making any investment. YuClawLab, its contributors, and affiliates accept no liability for any losses arising from use of this software. Long-form versions: `DISCLAIMER.md` and `docs/methodology/backfill.md`. Released under the MIT License.

YUCLAW User Guide · written for the v5.0 series · the repository README and docs/ are authoritative where this guide and newer releases differ.

A Appendix — quick reference card

Print this page and keep it near the terminal.

COMMANDS

```
yuclaw why TICKER
yuclaw replay TICKER --date DATE
yuclaw replay-lab
yuclaw validation
yuclaw verify TICKER --date DATE
yuclaw events --ticker SU --since DATE *
yuclaw lens canada --lens XEG *
yuclaw export --lens GDX --format csv *
yuclaw memo --ticker SU --days 30 *†
yuclaw brief · watch add · profile show
```

* ships in v5.1; run `python3 -m v3.cli` from a main checkout meanwhile † needs full local deployment

SURFACES

```
yuclawlab.github.io/yuclaw-brain
  /validation_lab.html
  /canada_resources.html
  /etf_evidence.html
  /todays_evidence.html
  /replication.html
github.com/YuClawLab/yuclaw-brain
pypi.org/project/yuclaw
Telegram: @yuclaw_signals (07:35 MT)
```

CITE YUCLAW

"YUCLAW v5.0.0, data through <date>, commit <hash>" — exact values ship in every evidence-packet METADATA.json.

The one habit: follow the source links. Every event in every signal resolves to a SEC filing — the engine is built to show its work. Make it.