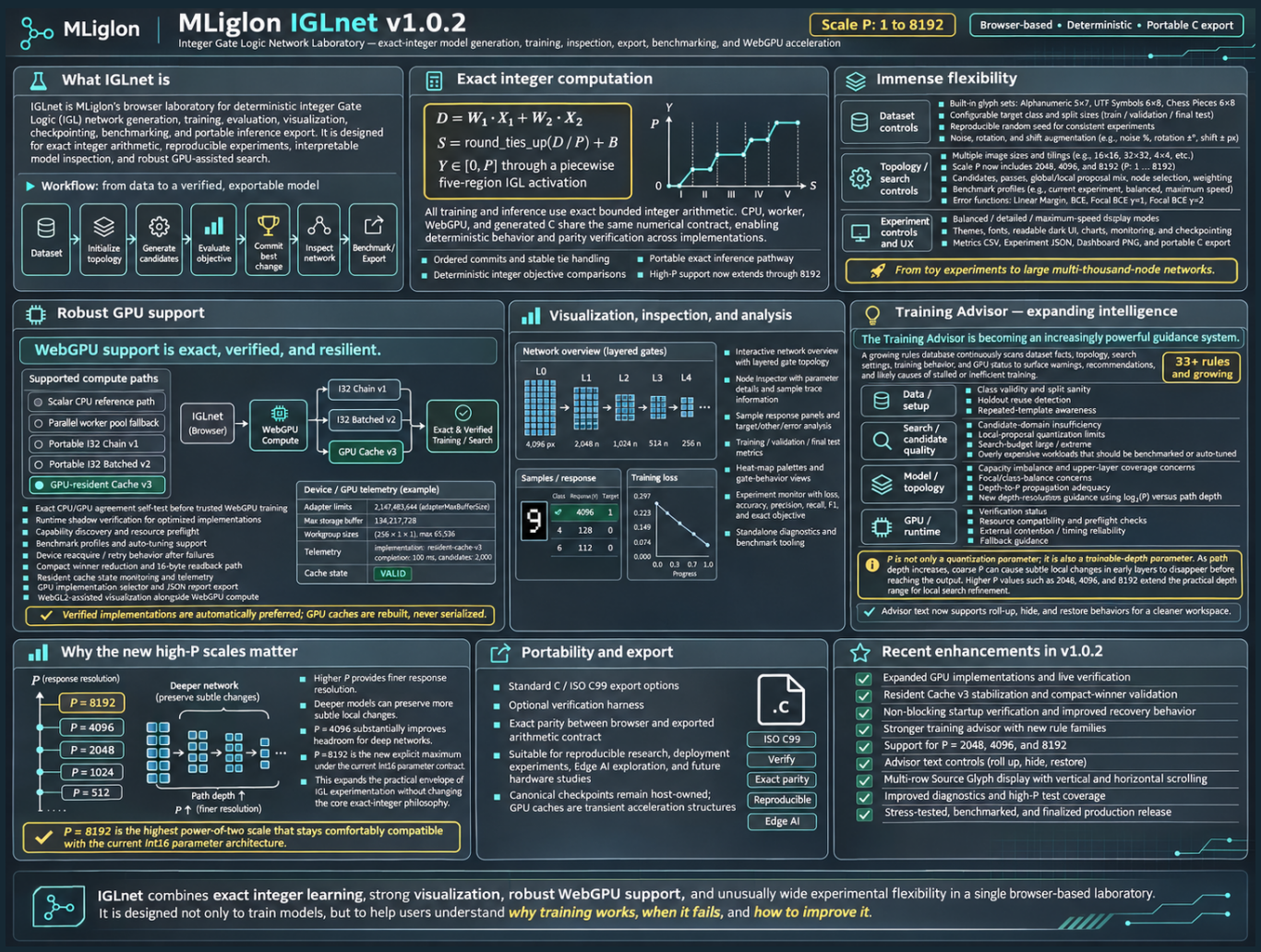
MLiglon

INTEGER GATE LABORATORY

IGLnet v1.0.2

Comprehensive User Guide



Production reference for the finalized IGLnet v1.0.2 release

Covers every current control, compute path, Advisor function, glossary term, and Advisor message.

About this guide

This manual documents the finalized IGLnet v1.0.2 browser laboratory as a reference work. It deliberately contains no tutorial exercises or worked training examples. Instead, it explains the program model, numerical contract, every current control and option, CPU and GPU execution, monitoring and export behavior, the Configuration Advisor, and the terminology used throughout the application.

Authority and scope
The final v1.0.2 production source tree is the authority for this guide. Control labels and option registries come from the shipped HTML and registries; Advisor messages come from advisor-rules.json and advisor.js; numerical and backend descriptions come from the executable code and engineering documentation.

Document conventions

Convention	Meaning
Control names	UI labels are shown in bold or in the Control Reference tables.
P	The global integer scale used for inputs, parameters, activations and output resolution.
Training / Validation / Final Test	Capitalized names refer to the three frozen generated dataset splits.
CPU / worker / WebGPU	Different scheduling implementations that are required to preserve the exact accepted integer result.
Blocking Advisor rule	A rule that prevents training until the condition is corrected; severity color alone does not imply blocking.

Version coverage

This edition covers the complete v1.0.2 feature set, including the three high-P scales 2048, 4096 and 8192; Portable I32 Chain v1; Portable I32 Batched v2; GPU-resident Cache v3; the 33-rule Training Advisor; Advisor roll-up/hide/restore controls; multi-row source glyph selection; and the final production checkpoint/export behavior.

Contents

About this guide 2

Document conventions . . . 2

Version coverage . . . 2

Contents 3

IGLnet at a glance 7

Part I - Product model and numerical foundations 8

1. Product overview . . . 8

2. Exact integer node contract 9

2.1 Supported scale P values . . . 9

2.2 Pixel normalization . . . 9

3. Loss, classification and strict adoption 10

3.1 Weighting . . . 10

3.2 Adoption rule . . . 10

4. Network construction and topology 11

4.1 Capacity and depth . . . 11

5. Dataset identity and evaluation discipline 12

5.1 Final Test . . . 12

5.2 Best-validation checkpoint . . . 12

Part II - Interface and control reference 13

6. Main workspace map . . . 13

7. Header, status and Appearance 14

7.1 Appearance options . . . 14

8. Dataset panel 15

8.1 Source and split controls . . . 15

8.2 Augmentation controls 16

8.3 Noise formula . . . 16

8.4 Source glyph selector 17

8.5 Installed glyph sets 18

9. Model & Search panel 19

9.1 Model and compute controls . . . 19

9.2 Search controls **20**

9.3 Node-selection strategies **21**

9.4 Candidate parameter search **22**

10. Training command bar and experiment state **23**

Part III - Training Advisor **24**

11. Advisor purpose and architecture . . . 24

11.1 Severity model . . . 24

12. Advisor controls and presentation modes **25**

13. Advisor derived search facts **26**

14. Advisor rule families **27**

15. Explain stalled training **28**

Part IV - Visualization, inspection and monitoring **29**

16. Samples panel . . . 29

17. Integer gate network view **30**

17.1 Network controls . . . 31

17.2 Gate-behavior interpretation **32**

17.3 Navigation . . . 32

18. Node Inspector **33**

19. Experiment Monitor **34**

19.1 Reported metrics . . . 35

19.2 History and sample responses **36**

20. Runtime event log and bottom actions **37**

Part V - CPU and GPU compute functionality **38**

21. Canonical scalar CPU backend . . . 38

22. CPU worker pool **39**

22.1 Worker correctness boundary . . . 39

23. WebGPU architecture **40**

23.1 WebGPU implementations . . . 40

23.2 Verification and runtime protection 41

23.3 Compact winner and Resident Cache v3 42

24. GPU capabilities dialog 43

24.1 GPU capability and telemetry fields 44

24.2 WebGL2 vs. WebGPU 45

Part VI - Benchmarking, diagnostics, persistence and export 46

25. Benchmark profiles . . . 46

26. Full diagnostics 47

27. Checkpoints and exact resume 48

28. Portable C export 49

28.1 C numerical and memory contract 50

29. Metrics and experiment exports 51

30. Display recovery controls 52

31. Browser, security and deployment boundaries 53

Part VII - Exhaustive glossary 54

Appendix A - All current Training Advisor messages 63

Appendix A-2 - Explain stalled training diagnosis messages 80

Appendix B - Exhaustive control and option inventory 81

B.1 Header and Appearance . . . 81

B.2 Dataset . . . 82

B.3 Model, search and training . . . 83

B.4 Visualization, monitoring and exports . . . 84

B.5 GPU, Benchmark, Export and Recovery dialogs . . . 85

Appendix C - Installed source glyph contents 86

C.1 Alphanumeric 5x7 . . . 86

C.2 UTF Symbols 6x8 . . . 86

C.3 Chess Pieces 6x8 . . . 86

Appendix D - v1.0.2 verification summary **87**

Document end 87

IGLnet at a glance

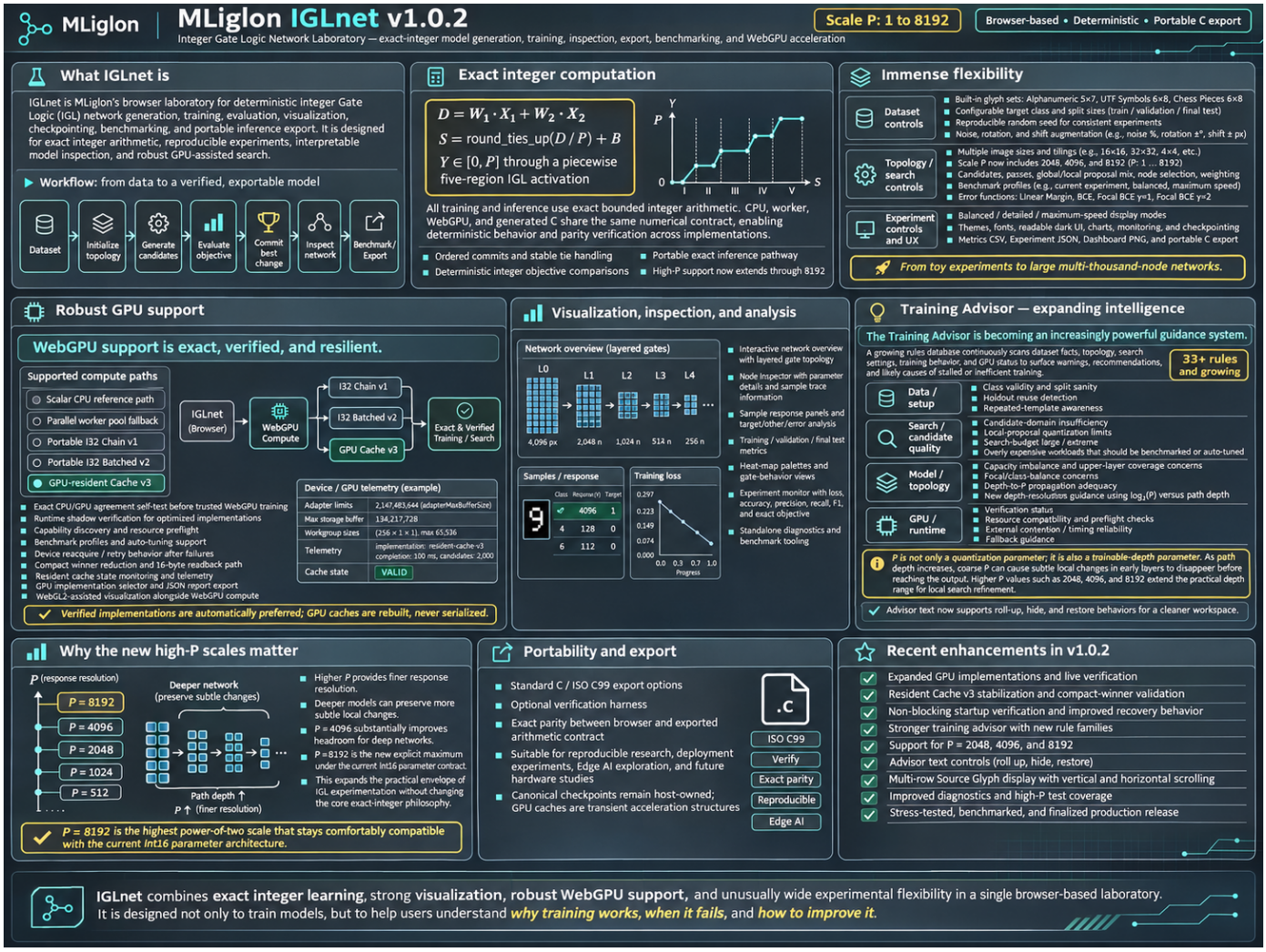


Figure 1. High-density overview of IGLnet v1.0.2 capabilities and architecture.

IGLnet is a self-contained browser laboratory for deterministic Integer Gate Logic network experiments. The application combines exact integer model arithmetic, reproducible generated datasets, strict greedy parameter search, layered visual inspection, multiple compute backends, a rules-based configuration/training Advisor, checkpointing, diagnostics, benchmark reporting, and portable C inference export.

Part I - Product model and numerical foundations

1. Product overview

IGLnet operates entirely in the browser. It has no server-side training service, remote model store, account database, or runtime network dependency. The dataset, model, random state, training history and evaluation history are canonical host-owned state. CPU workers and GPU buffers are execution accelerators; they do not replace the canonical experiment.

The central research model is an Integer Gate Logic network built from independent two-input computed nodes. Each node applies an exact bounded integer transform. Candidate parameter triples are proposed for selected nodes, scored against the full Training split, and committed only when the exact objective becomes strictly lower. This strict acceptance rule makes the committed Training objective non-increasing while the objective definition remains unchanged.

Core design principle
Different compute backends may schedule work differently, but they are expected to agree on the integer arithmetic and accepted model changes. WebGPU implementations must pass scalar-reference verification before they can train the canonical model.

1.1 Functional areas

Area	Purpose
Dataset generation	Select fixed bitmap glyph sources, target class, split sizes, seed and geometric/noise augmentation.
Model construction	Choose integer scale P and tiling; initialize a deterministic topology with independent node parameters.
Search / training	Choose passes, node coverage, candidate count, proposal mix, weighting and loss; accept only strict improvements.
Inspection	Examine generated samples, network topology, node parameters, actual sample traces, response heat maps and metrics.
Advisor	Continuously derive facts from data/model/search/training/GPU state and match them against a versioned rule database.
Compute	Run canonical scalar CPU, persistent CPU workers, or verified WebGPU implementations.
Persistence / reporting	Save checkpoints and export metrics CSV, experiment JSON, dashboard PNG, GPU reports and benchmark reports.
Inference export	Generate a no-heap ISO C99 inference package using the same exact integer node contract.

2. Exact integer node contract

For scale P, every input and node output is bounded to 0..P. Node parameters W1 and W2 are bounded to -2P..2P, and bias B is bounded to -P..3P. Products and the dot sum are widened before normalization.

```
D = W1*X1 + W2*X2 S = round_ties_toward_positive_infinity(D/P) + B
```

The activation maps S through five regions: S<=0 gives 0; 03P gives P. The products are summed before the single division/round operation. This operation order is part of the model contract.

2.1 Supported scale P values

P values	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192
Maximum P	8192 is the explicit maximum under the current signed Int16 parameter storage contract.
P=8192 boundaries	|W| <= 16384; B <= 24576; worst |D| = 4P^2 = 268,435,456, safely within signed 32-bit arithmetic.

Why P also affects trainable depth
P controls response resolution. In deeper chains, fine local changes can be lost to integer rounding before they reach the root. The Advisor therefore derives a depth-resolution margin log2(P) - maximum path depth and warns when lower-layer refinement is likely to be marginal or severely limited.

2.2 Pixel normalization

Source grayscale g in 0..255 is quantized to the model scale by round(g*P/255). Fixed bitmap assets are used rather than browser font rasterization, preserving source identity and reproducibility.

3. Loss, classification and strict adoption

The loss registry contains four objectives. All candidate comparison is exact and stable across the reference CPU and verified accelerated implementations.

Loss	Registry ID	Meaning
Linear Margin	margin-linear-v1	Exact threshold-oriented integer objective. Class 0 numerator max(0,20Y-9P); class 1 numerator max(0,11P-20Y).
Binary Cross Entropy	bce-v1	Precomputed fixed-point BCE with epsilon=1e-6 and table scale 512.
Focal BCE gamma=1	focal-bce-g1-v1	Fixed-point focal BCE with moderate emphasis on difficult examples.
Focal BCE gamma=2	focal-bce-g2-v1	Fixed-point focal BCE with stronger emphasis on difficult examples.

Classification is separate from loss. A sample is predicted as target/class 1 when 2Y >= P. The normalized response Y/P is a bounded response measure and is not a calibrated probability.

3.1 Weighting

Class balanced gives the two classes equal aggregate influence even when their sample counts differ. **Equal samples** gives every sample equal weight and therefore allows the majority class to dominate the aggregate when the split is imbalanced.

3.2 Adoption rule

Candidate ordering provides a stable tie break among proposed alternatives, but the current node is retained on a tie. A node update is committed only when its exact candidate objective is strictly lower than the current exact Training objective.

4. Network construction and topology

The original W x H canvas can be tiled 1x1, 2x2 or 4x4. Adjacent columns are reduced first, then adjacent rows, alternating until one root remains. Odd edges preserve the real source in X1 and inject a zero source into X2. Every computed node owns independent W1, W2 and B.

Source canvas	8x8, 10x10, 16x16, 32x32
Tiling	1x1, 2x2, 4x4
Model state	Independent parameters at every computed node
Pass direction	Cycles output-first, random, input-first
Partial pass	Checkpoint preserves exact node order/cursor and RNG state

A selected node affects one path to the root. Candidate scoring packages the selected-node source values and the unaffected sibling/slot/parameters of every ancestor. This affected-path representation is central to both CPU parallelism and the GPU kernels.

4.1 Capacity and depth

Larger tiling increases computed-node count and path depth. Capacity can help representation, but it also raises the search workload and can reduce the visibility of small local parameter changes at low P. The Advisor monitors both capacity relative to dataset diversity and depth relative to P.

5. Dataset identity and evaluation discipline

Each installed source set has fixed dimensions, version information and a declared SHA-256 identity. Generated splits are frozen at dataset generation time. Noise is sampled and clipped during generation; the model does not see a live augmentation transform during candidate scoring.

Training and Validation are independent generated splits. With augmentation disabled, they can contain repetitions of the same fixed source templates, so identical aggregate metrics are expected and do not demonstrate generalization to unseen appearances. The program surfaces split identities, uniqueness and overlap to make this explicit.

5.1 Final Test

Final Test is sealed in the normal interface until explicit disclosure. Every disclosure records model identity/revision, parameter digest, Test identity and whether the same Test split had already been disclosed. Repeated Test inspection is flagged by the Advisor because it changes the interpretation of the holdout.

5.2 Best-validation checkpoint

The best Validation checkpoint is tracked separately from the current model. Restore best creates a new revision rather than silently rewriting history.

Part II - Interface and control reference

6. Main workspace map

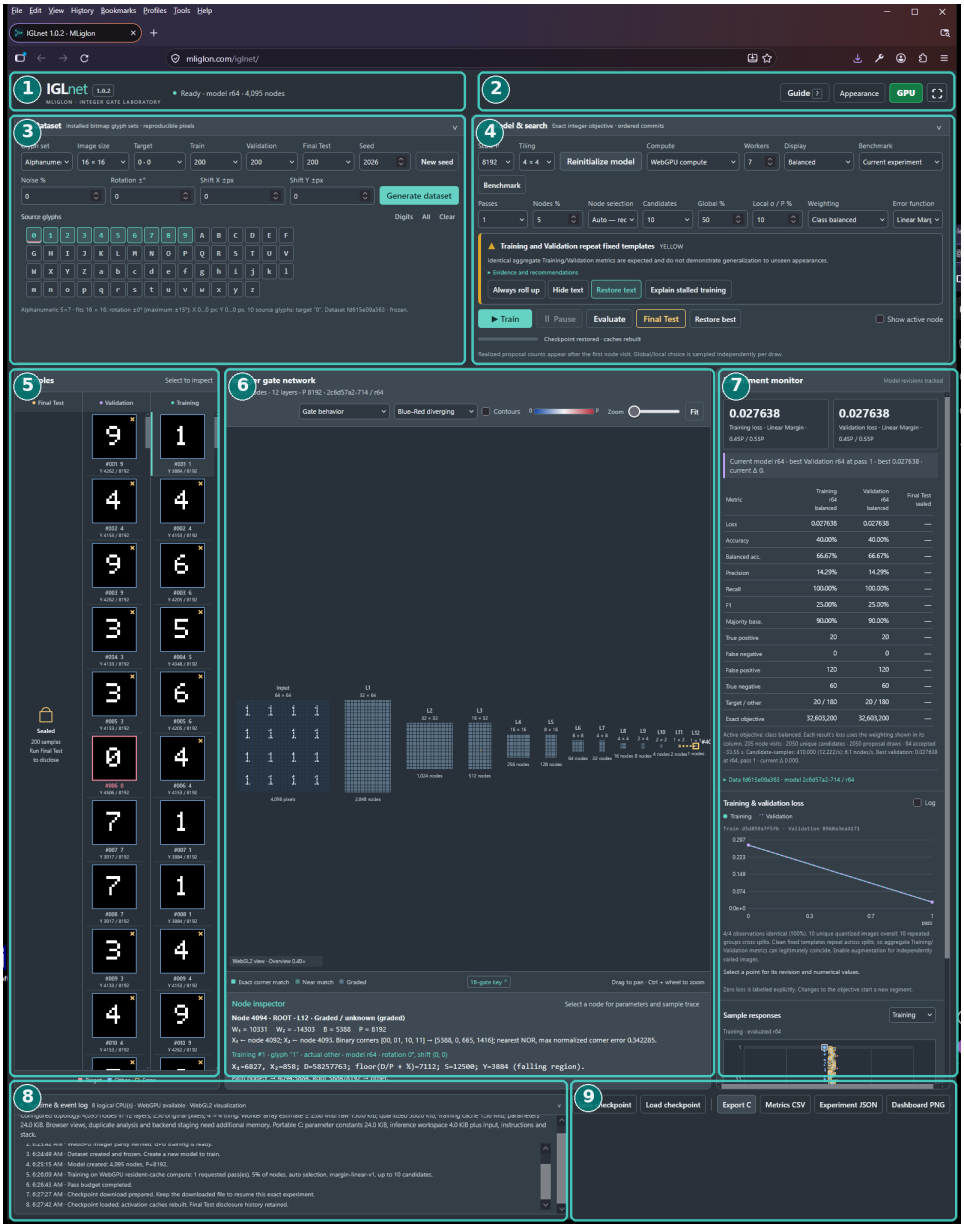


Figure 2. Main workspace. Numbered regions: 1 brand/status, 2 header actions, 3 Dataset, 4 Model & Search/Advisor, 5 Samples, 6 network/Node Inspector, 7 Experiment Monitor, 8 runtime log, 9 checkpoint/export actions.

The desktop layout is a dense, resizable laboratory workspace. Collapsible Dataset and Model & Search panels occupy the top. Samples, network inspection and experiment monitoring occupy the main analysis row. Runtime events and persistence/export controls remain available at the bottom.

7. Header, status and Appearance

The header contains the MLiglon IGLnet wordmark, current readiness/model status, Guide access, Appearance controls, GPU status/dialog access and fullscreen control.

Control	Options / range	Function and operational meaning
Guide	Button; ? shortcut	Opens the built-in reference guide and keyboard shortcut list.
Appearance	Popover	Contains theme, typeface, text-size and desktop panel-width controls. Preferences are presentation-only.
GPU	Status button / dialog	Displays readiness state and opens WebGPU/WebGL capabilities, implementation selection, verification, telemetry and diagnostics.
Fullscreen	Toggle	Requests browser fullscreen when permitted.
Status	Read-only status line	Reports readiness, model revision/state and high-level runtime state.

7.1 Appearance options

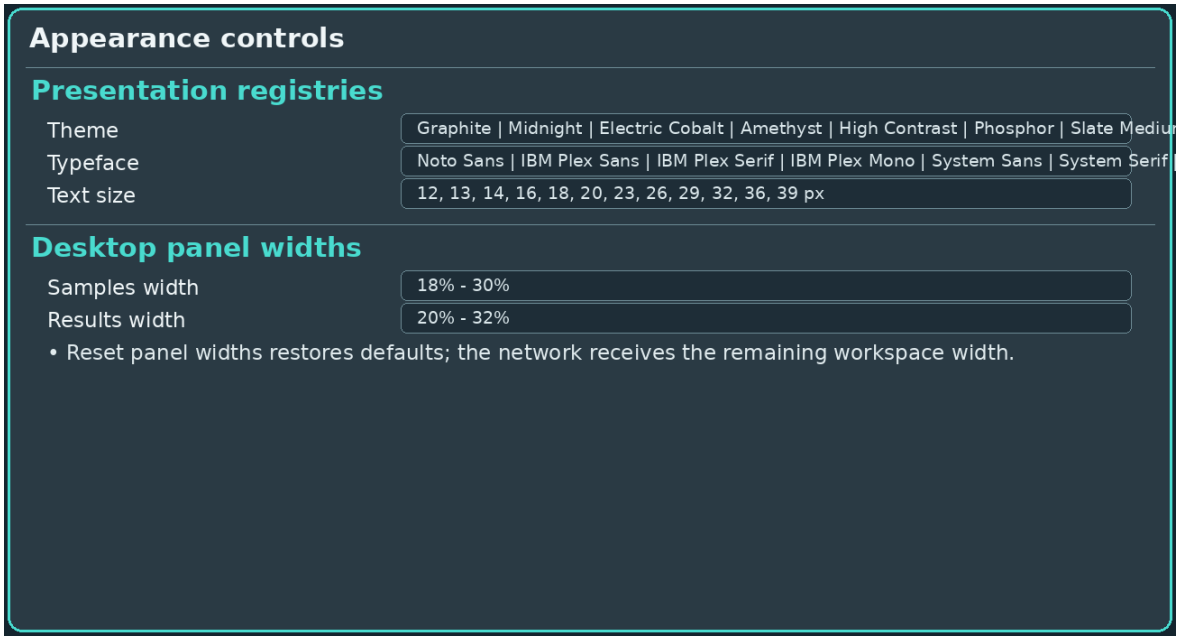


Figure 3. Appearance and desktop panel-width controls.

Control	Options / range	Function and operational meaning
Theme	16 themes	Graphite, Midnight, Electric Cobalt, Amethyst, High Contrast, Phosphor, Slate Medium, Teal Medium, Sepia Medium, Ocean Blue, Evergreen, Warm Sand, Glacier Steel, Ivory & Gold, Rose Quartz, Technical Paper.
Typeface	11 preferences	Noto Sans, IBM Plex Sans/Serif/Mono, system sans/serif/mono, Arial/Helvetica, Georgia, Noto Serif, Noto Mono. Named fonts use locally installed fonts or fallbacks; no font download occurs.
Text size	12 values	12, 13, 14, 16, 18, 20, 23, 26, 29, 32, 36, 39 px.
Samples width	18%-30%	Desktop width assigned to the Samples panel.
Results width	20%-32%	Desktop width assigned to Experiment Monitor. Network receives remaining workspace width.
Reset panel widths	Button	Restores default desktop proportions.

8. Dataset panel

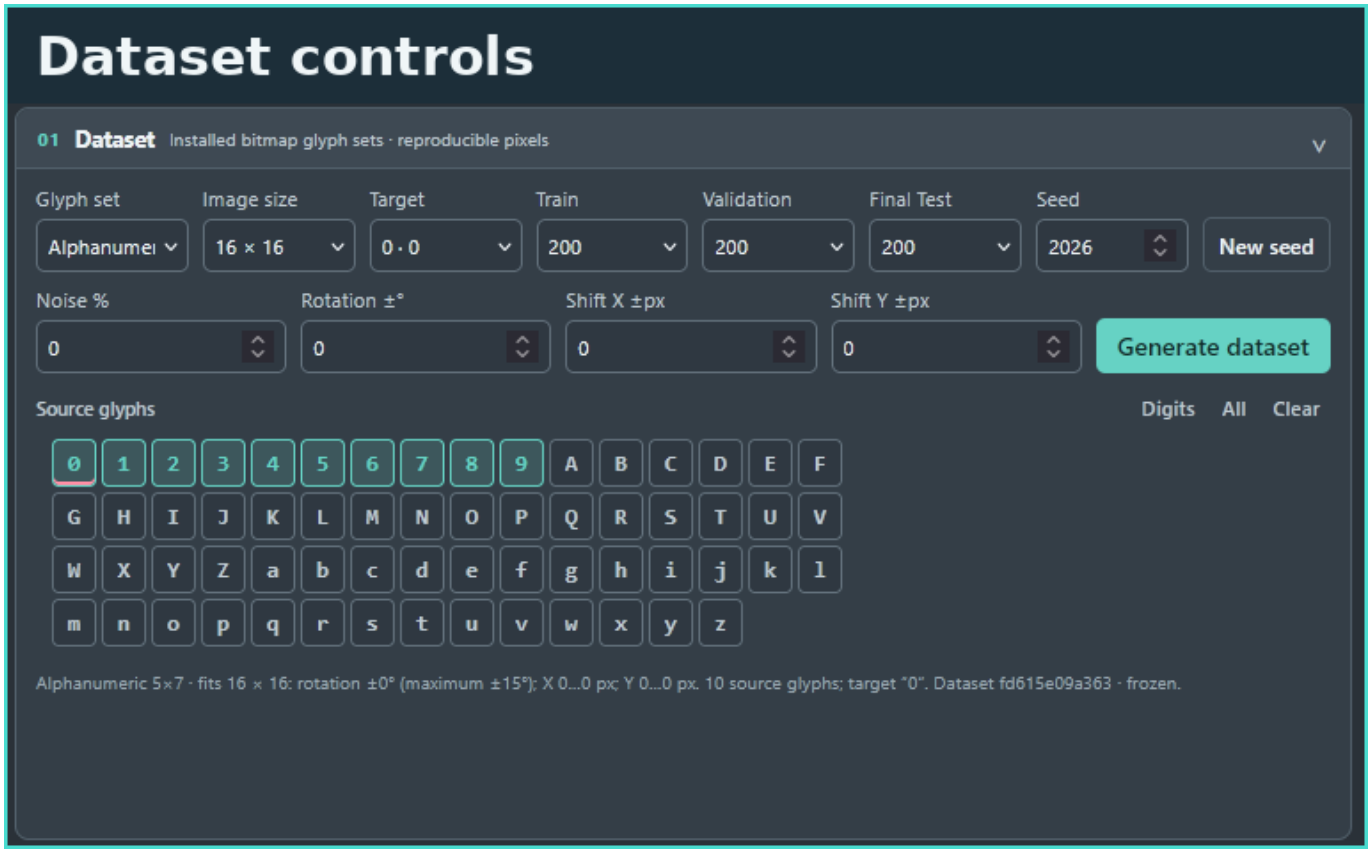


Figure 4. Dataset controls in the finalized v1.0.2 interface.

Dataset controls define the source glyph family, generated canvas, binary target class, split sizes, reproducible seed, augmentation envelope and the source glyph pool. Dataset regeneration starts a fresh experiment because the model depends on the quantized frozen data.

8.1 Source and split controls

Control	Options / range	Function and operational meaning
Glyph set	Alphanumeric 5x7; UTF Symbols 6x8; Chess Pieces 6x8	Selects the installed fixed bitmap source set and its provenance.
Image size	8x8, 10x10, 16x16, 32x32	Generated canvas dimensions. Native glyph-cell dimensions remain fixed within the selected source set.
Target	Selected glyph from source pool	Defines positive class 1. Other selected source glyphs form class 0.
Train	10, 100, 200, 1000	Number of generated Training images. Candidate scoring always evaluates the complete Training split.
Validation	10, 100, 200, 1000	Number of generated Validation images used for model selection and monitoring.
Final Test	10, 100, 200, 1000	Number of generated sealed Final Test images.
Seed	0..4,294,967,295	Reproducible 32-bit experiment seed.
New seed	Button	Chooses a fresh seed without changing the current experiment until Generate dataset is invoked.

8.2 Augmentation controls

Control	Options / range	Function and operational meaning
Noise %	0..100	Gaussian pixel noise. Standard deviation before clipping is 10.2 x intensity percent. Noise is frozen at generation time.
Rotation +/- degrees	0..15	Uniform integer-degree rotations. Effective bounds can tighten to prevent clipping within the source-cell envelope and canvas.
Shift X +/-px	0..32	Horizontal integer-pixel translation; legal combinations are geometry-limited.
Shift Y +/-px	0..32	Vertical integer-pixel translation; legal combinations are geometry-limited.
Generate dataset	Button	Creates frozen Training, Validation and Final Test splits from the current dataset controls and source glyph selection.

Geometry-limited augmentation
The program derives safe rotation/translation limits from the selected source set and canvas. An Advisor message reports when the requested augmentation is reduced by geometry.

8.3 Noise formula

```
sigma = 1020 * intensityPercent / 100 pixel = round(clamp(transformedPixel + sigma*Normal(0,1), 0, 255))
```

8.4 Source glyph selector

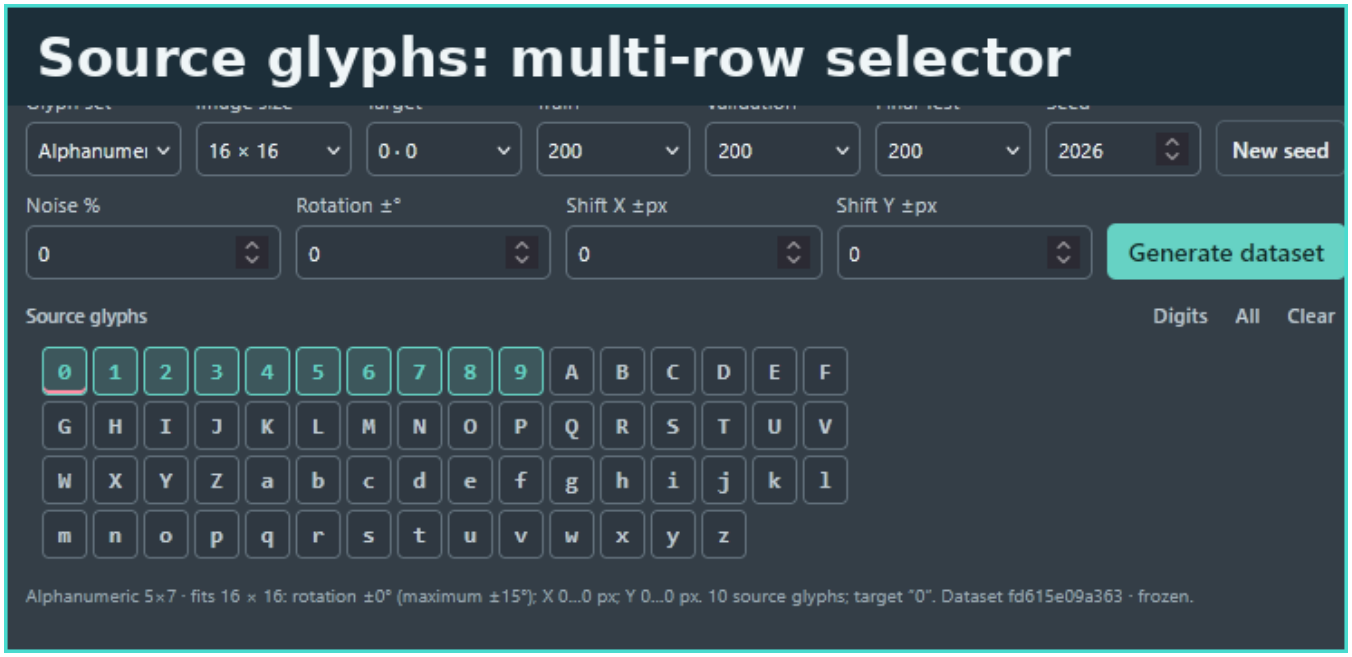


Figure 5. Multi-row Source Glyph selector introduced in Hotfix 8; both vertical and horizontal scrolling are retained when required.

Control	Options / range	Function and operational meaning
Individual glyph buttons	Every glyph in the installed source set	Toggles inclusion in generated data. Target must remain selected, and at least one non-target glyph is required for a valid binary objective.
Set defaults	Button	Restores the selected source set default pool: digits for Alphanumeric; all symbols/pieces for the other installed sets.
All	Button	Selects the complete source set.
Clear	Button	Clears the source selection subject to validity controls.

Alphanumeric contains 62 glyphs; the finalized selector uses a multi-row desktop grid so the full set can often be viewed simultaneously. The viewport maintains two-axis overflow so larger text sizes and narrow windows remain usable.

8.5 Installed glyph sets

Set	Dimensions	Count	Pixel mode	Default target / selection
Alphanumeric 5x7	5x7	62	Binary	Target 0; digits selected by default
UTF Symbols 6x8	6x8	36	Binary	Target Star; all symbols selected
Chess Pieces 6x8	6x8	12	Grayscale	Target White King; all pieces selected

The glyph labels for the symbol and chess sources are Unicode names, but the training pixels come from fixed project bitmap arrays. Browser font rasterization is never used to create training data.

9. Model & Search panel

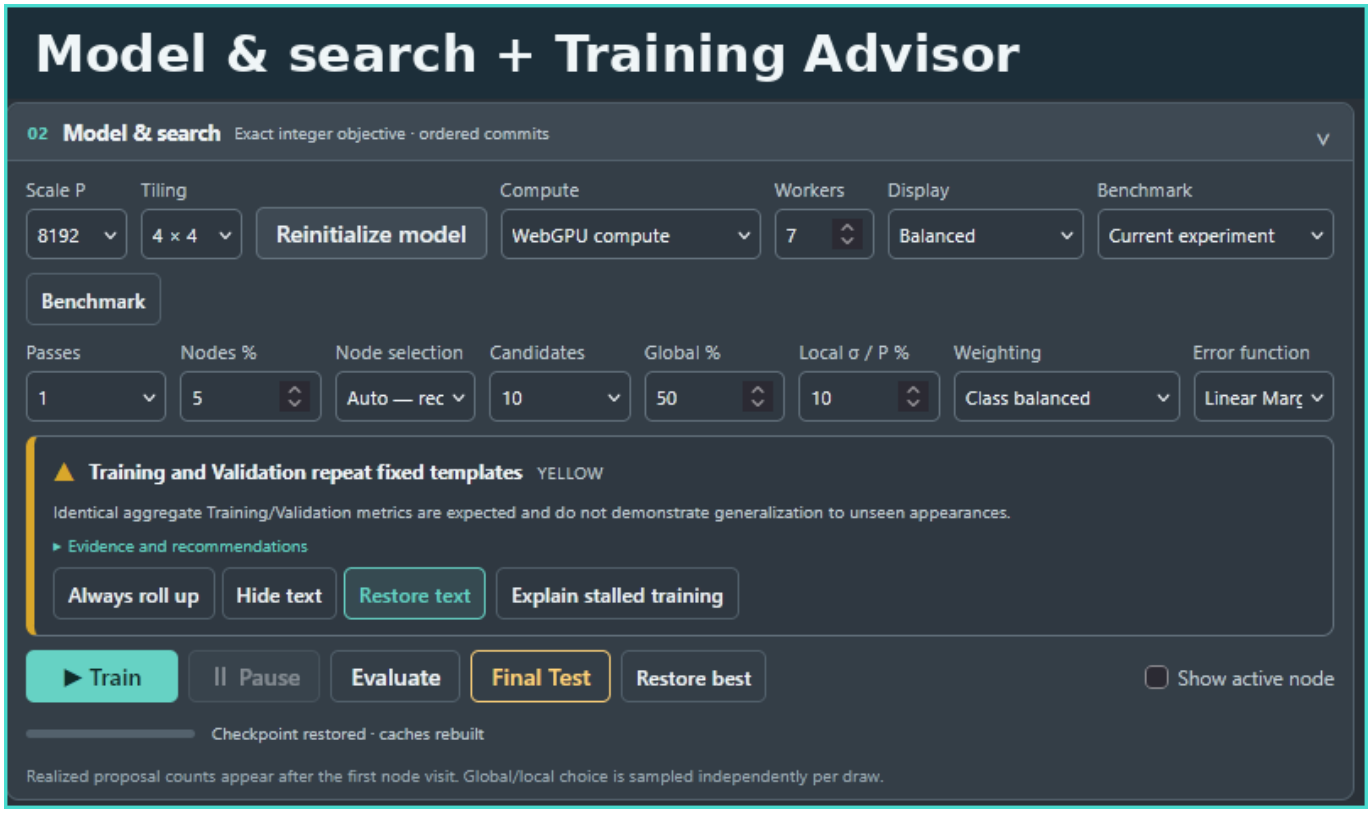


Figure 6. Model construction, compute, search controls and the live Training Advisor.

9.1 Model and compute controls

Control	Options / range	Function and operational meaning
Scale P	1,2,4,8,16,32,64,128,256,512, 1024,2048,4096,8192	Global integer scale. Changing P requires explicit model reinitialization. P=8192 is the maximum under the current Int16 parameter contract.
Tiling	1x1, 2x2, 4x4	Replicates the source canvas along both axes; every computed node has independent parameters.
Initialize model	Button	Builds a fresh network for the frozen dataset using the selected P and tiling.
Compute	WebGPU compute; CPU worker pool; Single CPU reference	Selects the training scoring backend. WebGPU remains gated by verification.
Workers	1..32	Requested persistent CPU worker count when the worker-pool backend is active.
Display	Maximum speed; Balanced; Detailed	Controls visualization/update throttling. It does not change the numerical objective.
Benchmark	Small; Medium; Large/GPU-oriented; Current experiment	Chooses the disposable benchmark workload profile.
Benchmark button	Button	Runs backend comparison without mutating model, RNG or experiment history.

9.2 Search controls

Control	Options / range	Function and operational meaning
Passes	1, 10, 100, 1000	Number of requested passes for the Train action. Pass direction cycles output-first, random and input-first.
Nodes %	1..100	Unique node fraction selected per pass, rounded upward.
Node selection	Auto; Uniform; Layer-balanced	Auto uses Uniform for 1x1/2x2 and Layer-balanced for 4x4.
Candidates	1,2,5,10,100,200	Requested unique changed W1/W2/B triples evaluated per visited node.
Global %	0..100	Probability that each proposal is sampled from the legal global parameter range.
Local sigma / P %	1..100	Local Gaussian standard deviation expressed as a percentage of P.
Weighting	Class balanced; Equal samples	Controls exact objective aggregation across target and non-target classes.
Error function	Linear Margin; BCE; Focal BCE gamma=1; Focal BCE gamma=2	Versioned exact candidate-ordering objective.

Realized proposals
Global/local selection is a Bernoulli draw for each proposal. A particular node need not receive exactly the configured global percentage. Candidate statistics report requested candidates, unique scored triples, duplicates/no-ops and one-step local fallbacks.

9.3 Node-selection strategies

Strategy	Behavior	Advisor relevance
Auto - recommended	Uniform for 1x1/2x2; Layer-balanced for 4x4.	Designed to reduce decision-side starvation without changing simpler topologies.
Uniform - legacy v1.0.x	Samples uniformly from all computed nodes.	Low Nodes % on deep/4x4 models can leave upper layers untouched for multiple passes.
Layer-balanced	Distributes selected-node budget across computed layers while preserving the total visit budget.	Cannot visit every layer in a pass if the selected-node budget is smaller than the layer count.

Visit order and selection strategy are distinct. After selection, the pass order cycles output-first, random and input-first. Ordered commits preserve deterministic training semantics even when candidate/sample scoring is parallel.

9.4 Candidate parameter search

Each candidate is a changed W1/W2/B triple. The legal finite domain at scale P contains (4P+1)^3 - 1 changed triples relative to the current value. At extremely small P, the requested candidate count can exceed the complete available domain; the Advisor detects this before training.

Local proposals are Gaussian perturbations of the current triple. The absolute local standard deviation is P x (local sigma/P fraction). If that absolute scale falls below half an integer step and local proposals are a material fraction of the search, many draws round back to the current values and rely on bounded one-step fallback mutations.

Global proposals explore the full legal parameter range. Maintaining a nonzero global fraction is particularly useful when low-P/deep-path rounding makes fine local changes invisible at the root.

10. Training command bar and experiment state

Control	Options / range	Function and operational meaning
Train	Button	Starts the requested passes using the selected backend and current search configuration.
Pause	Button	Requests pause; it takes effect at a committed node boundary.
Evaluate	Button	Refreshes Training and Validation metrics for the current model without changing parameters.
Final Test	Button	Explicitly discloses the sealed Final Test for the current frozen model and records the exposure.
Restore best	Button	Restores the earliest checkpoint with the best observed Validation loss as a new model revision.
Show active node	Checkbox	When enabled, visualization follows/highlights the currently active training node subject to display throttling.

The line immediately below the training buttons reports checkpoint restore state, cache state and other experiment-transition information. Realized proposal counts appear after the first node visit because proposal generation is stochastic and independent per draw.

Part III - Training Advisor

11. Advisor purpose and architecture

The Configuration Advisor is a continuously evaluated, versioned rules system. It derives facts from the current dataset, model topology, search settings, training history, candidate-generation statistics and GPU state. Those facts are matched against a data-only rule database. Rule text is not executed with eval or dynamic code generation.

Each rule contains an ID, priority, severity, an independent blocking flag, structured predicates, evidence, consequence and recommendations. Rules can expose safe, explicit Apply actions. Severity and blocking are intentionally separate: a red warning can remain nonblocking for research, while a red correctness/resource rule can block training.

11.1 Severity model

Color	Meaning	Blocking implication
White	Not yet assessable, informational or insufficient evidence.	Usually nonblocking.
Green	Healthy or useful positive signal.	Nonblocking.
Yellow	Caution, efficiency/generalization/search-quality concern.	Normally nonblocking.
Red	Severe risk, repeated stall, or correctness/resource problem.	May be blocking or nonblocking; consult the rule.

12. Advisor controls and presentation modes

Control	Options / range	Function and operational meaning
Evidence and recommendations	Expandable details	Shows matched-rule evidence, consequence and recommendations.
Always roll up	Persistent mode	Closes evidence/details, suppresses the stall report and clamps the summary so Advisor expansion cannot consume the Model & Search panel vertically.
Hide text	Persistent mode	Hides Advisor summary/evidence/report text while retaining the status heading and rule evaluation.
Restore text	Persistent mode	Returns to normal user-controlled expansion.
Explain stalled training	Button	Generates a ranked diagnosis using coverage, candidate health, depth/P margin, quantization/local-scale facts, plateau history and capacity evidence.
Apply recommendation	Conditional button	Appears for supported safe actions, such as switching to Layer-balanced selection.

Presentation does not change evaluation
Always roll up, Hide text and Restore text modify only presentation. They do not change derived facts, matched rules, severity, blocking state or training semantics.

13. Advisor derived search facts

Derived fact	Definition	Interpretation
maximum propagation depth	max(0, layerCount-1)	Longest computed-node path from earliest layer to the root.
P resolution bits	log2(P)	Binary-like resolution measure used to compare P with path depth.
depth-resolution margin	log2(P) - maximum path depth	<= -2 severe; -1..0 marginal; >=1 no depth/P warning.
absolute local sigma	P x localSigmaFraction	Local proposal scale measured in integer parameter steps.
candidate domain size	(4P+1)^3 - 1	Finite upper bound on distinct changed parameter triples.
candidate samples/pass	Training samples x selected nodes x candidates	Lower-bound workload measure before path length is considered.
candidate samples/run	candidate samples/pass x requested passes	Search-budget indicator used by large/extreme workload rules.

P is a trainable-depth parameter
The depth/P rules do not claim that a deep low-P model cannot train. They specifically warn that fine lower-layer local changes can disappear through repeated rounding before reaching the root. Upper layers and sufficiently large global proposals can still improve the model.

14. Advisor rule families

Family	Representative concerns
Experiment validity	Dataset/model availability; missing target class; missing non-target class.
Coverage / selection	4x4 uniform low coverage; weak upper-layer coverage; last pass skipped decision-side layers; layer-balanced budget smaller than layer count.
Data / generalization	Repeated fixed templates; few target examples; severe noise; geometry-reduced augmentation; Final Test reuse.
Depth / quantization	Severe or marginal depth/P propagation; coarse P; sub-step local proposal scale.
Candidate quality	Finite-domain insufficiency; high duplicate/no-op rate; bounded-draw candidate shortfall; sparse candidates for large topology.
Search budget / progress	Large/extreme requested workloads; objective progress before accuracy moves; one-pass or repeated no-improvement.
Capacity / overfit	Possible overfit; low capacity; excess capacity.
Loss balance	Focal loss plus equal-sample weighting on an imbalanced split.
GPU	Unverified implementation; resource-preflight failure; externally contended timing.
Healthy	No known blocking configuration problem detected.

The complete current rule text is reproduced in Appendix A. The rule database contains 33 rules in v1.0.2.

15. Explain stalled training

Explain stalled training is a diagnosis layer in addition to the persistent 33-rule database. It ranks a compact set of possible causes using current facts. It can report upper-layer coverage inadequacy, degraded or healthy candidate generation, depth/P propagation limitation, coarse quantization, adequate depth/quantization margin, sub-step local proposal scale, persistent plateau, and insufficient completed passes to implicate capacity.

The report is diagnostic rather than an optimizer. Recommendations point to configuration dimensions that are already represented in the main controls: node-selection coverage, P, topology depth, global/local mix, local sigma and pass count.

All current diagnosis message templates are reproduced after the rule table in Appendix A.

Part IV - Visualization, inspection and monitoring

16. Samples panel

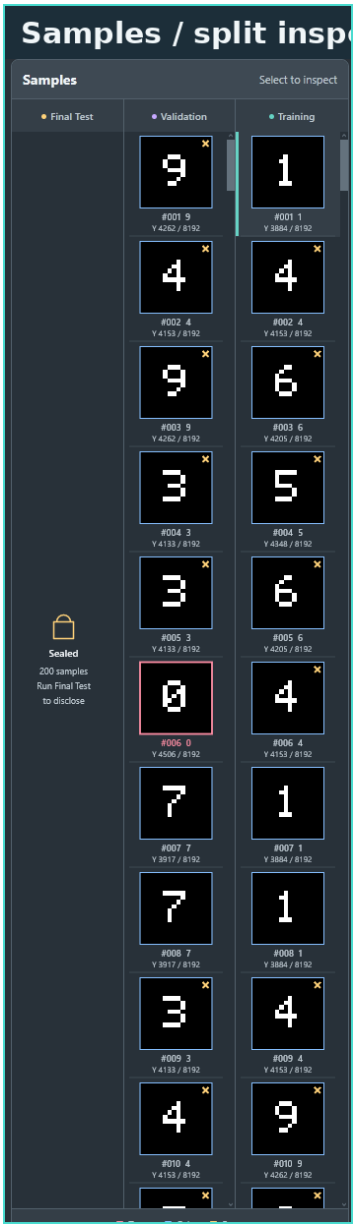


Figure 7. Samples panel with sealed Final Test indicator and Training/Validation thumbnails.

The Samples panel displays generated split images and their identities. Final Test remains visually sealed until explicit disclosure. Selecting a sample makes it available to the network visualization and Node Inspector for actual-output tracing.

Sample cards identify split membership and internal sample identity. Training and Validation can contain repeated source templates when augmentation is disabled, but their generated-split identities/order remain distinct.

17. Integer gate network view

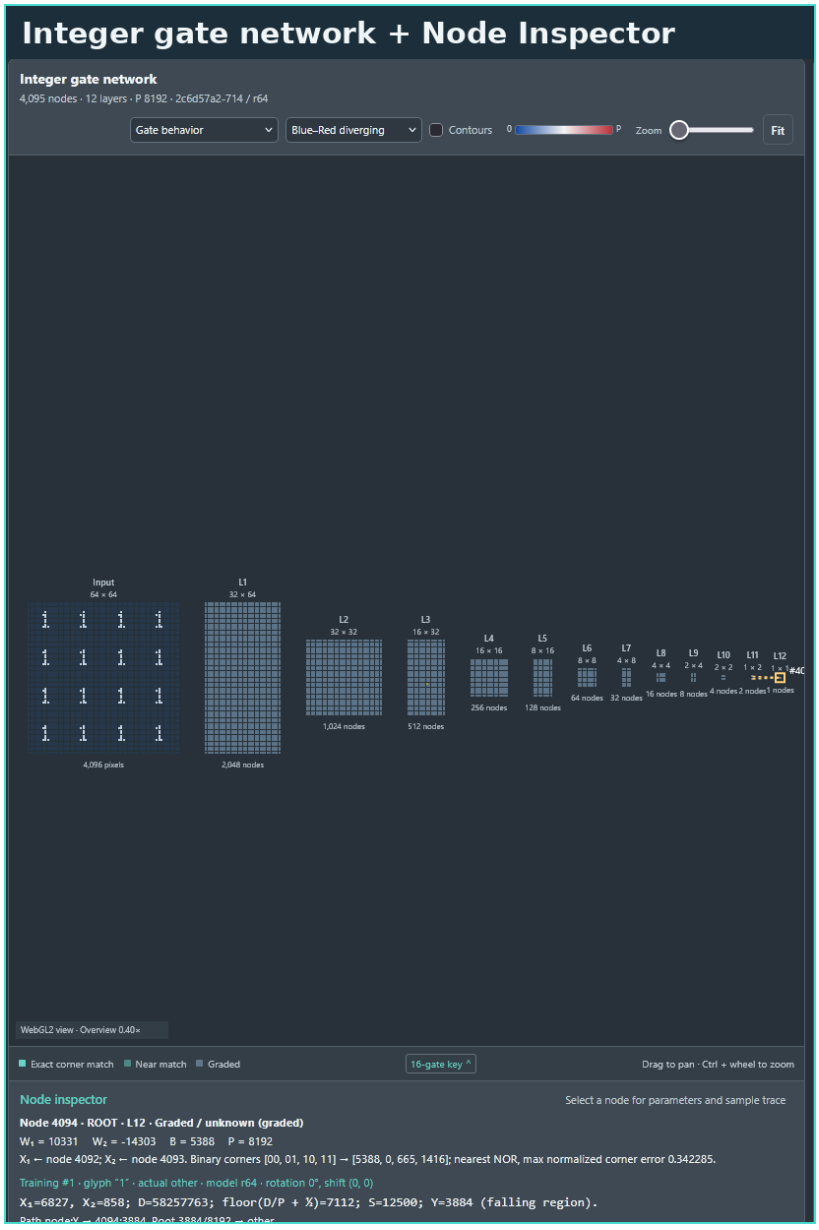


Figure 8. Layered network overview and Node Inspector at P=8192.

The network view presents the tiled input and computed layers in a single pan/zoom workspace. Layer labels and node counts expose topology depth and reduction structure. Nodes can be selected for parameter inspection and sample-trace interpretation.

17.1 Network controls

Control	Options / range	Function and operational meaning
Node coloring	Gate behavior; Sample output heat map	Switches between gate-character visualization and selected-sample output response.
Heat-map palette	Thermal; Topographical; Viridis; Plasma; Inferno; Magma; Turbo/Rainbow; Blue-Red diverging; Grayscale; High-contrast discrete	Presentation-only mapping of response values to colors.
Contours	Checkbox	Adds contour bands to heat-map presentation. Does not change node outputs.
Zoom	1x..30x, step 0.25	Controls network magnification.
Fit	Button	Fits the complete network into the available view.
16-gate key	Expandable key	Displays Boolean corner patterns/names used to interpret exact corner behavior.

17.2 Gate-behavior interpretation

Gate colors characterize the four binary input corners in order 00, 01, 10, 11. An exact corner match describes those corner responses, not every interior integer input pair. Nodes that do not exactly match a Boolean corner pattern can still be useful graded transforms.

At increasing zoom levels the network can reveal corner bits, short gate names and finally stored W1, W2 and B. The Node Inspector remains the authoritative textual presentation of a selected node regardless of zoom.

17.3 Navigation

Input	Function
Drag	Pan network
Ctrl + mouse wheel	Zoom at pointer
+ / -	Zoom focused network
0	Fit network
Left / Right arrow	Navigate child / parent
Up / Down arrow	Previous / next node or sample
Esc	Close active dialog

18. Node Inspector

The Node Inspector identifies the selected node, computed layer and gate classification, then reports W1, W2, B and P. For the selected sample it shows source/input values, the widened dot D, normalized/biased S, final Y and activation region. It also identifies the node-to-root path and sample class.

Inspection vs. model state
The inspector reads canonical model/sample state. Changing palettes, contours, zoom, fonts or themes does not change arithmetic or parameters.

19. Experiment Monitor

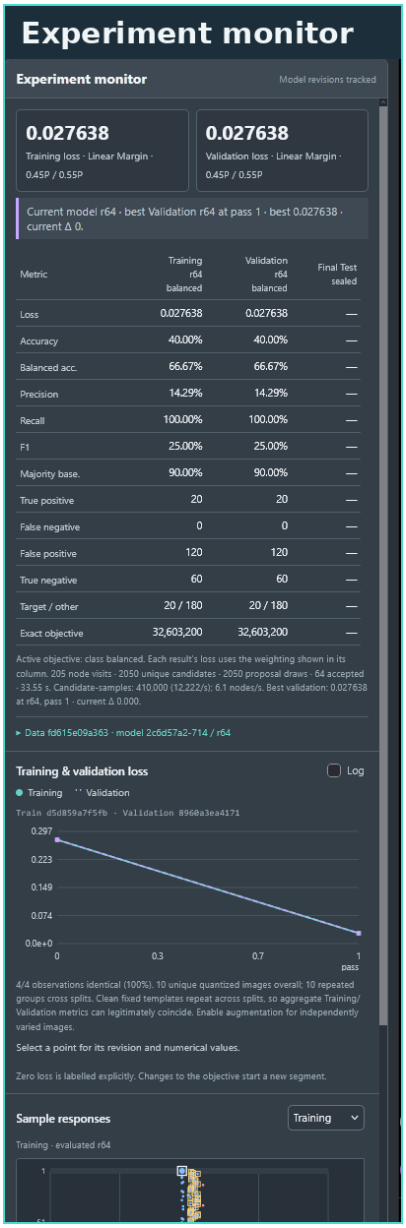


Figure 9. Experiment Monitor with exact loss, classification metrics, history chart and response analysis.

The Experiment Monitor tracks the current revision and the best Validation checkpoint, compares Training and Validation metrics, records Final Test only after disclosure, and visualizes training/validation loss history.

19.1 Reported metrics

Metric	Meaning
Loss	Normalized display of the active exact objective. Loss values from different objective definitions are not numerically comparable; objective-segment changes are marked.
Accuracy	Overall binary classification accuracy using 2Y>=P as target.
Balanced accuracy	Mean of target and non-target recall.
Precision	Target precision; undefined ratios are displayed appropriately.
Recall	Target recall.
F1	Harmonic mean of precision and recall when defined.
Majority baseline	Accuracy achieved by always selecting the majority class.
TP/FN/FP/TN	Binary confusion counts.
Target / other	Class sample counts.
Exact objective	Integer objective used for strict candidate adoption.

19.2 History and sample responses

Control	Options / range	Function and operational meaning
Log	Checkbox	Switches the loss-history y-axis to logarithmic presentation when appropriate.
Response dataset	Training; Validation; Final Test	Selects the split used by the response distribution plot; Final Test is subject to disclosure state.

Training loss is non-increasing only while the same objective definition is active and strict adoption is used. Validation loss and classification metrics can rise or fall. Changing the loss function creates a new objective segment rather than implying direct numerical continuity with the previous loss scale.

20. Runtime event log and bottom actions

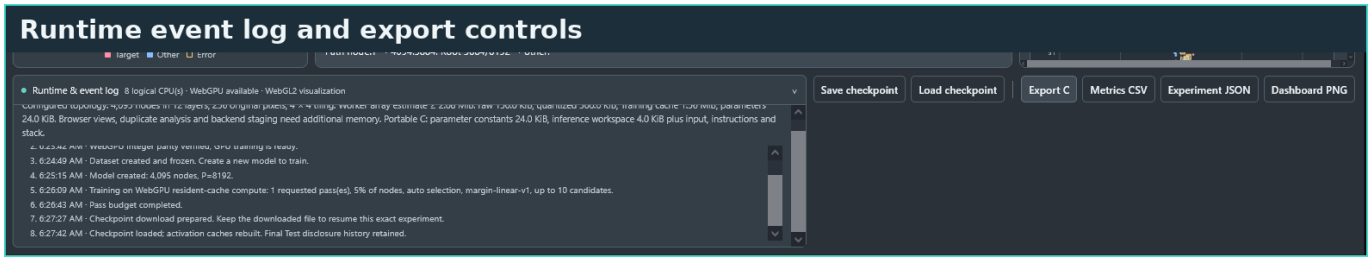


Figure 10. Runtime event log, checkpoint controls and export actions.

Control	Options / range	Function and operational meaning
Runtime event log	Scrollable text area	Reports important training, evaluation, backend, cache, checkpoint and disclosure events.
Save checkpoint	Button	Downloads the canonical experiment state needed for exact restoration.
Load checkpoint	Button + file selection	Imports and structurally validates a checkpoint before replacing the live experiment.
Export C	Button	Opens portable inference export configuration.
Metrics CSV	Button	Exports current/reportable experiment metrics as CSV.
Experiment JSON	Button	Exports the experiment manifest and relevant state/identity/report information.
Dashboard PNG	Button	Exports a dashboard image of the current monitoring state.

Part V - CPU and GPU compute functionality

21. Canonical scalar CPU backend

Single CPU reference is the canonical exact-integer implementation. It is intentionally simple and is used for correctness tests, bounded workloads, diagnostics and parity verification of optimized backends. It evaluates the same candidate/search semantics without worker or GPU scheduling.

Scalar execution is not assumed to be the fastest choice. Very small jobs can nevertheless favor it because worker/GPU dispatch overhead can exceed useful arithmetic. The program therefore includes local benchmark facilities rather than a fixed hardware-independent crossover rule.

22. CPU worker pool

The CPU worker-pool backend keeps persistent same-origin module workers. Candidate/sample work is partitioned and exact class totals are reduced deterministically. SharedArrayBuffer is not required.

The Workers control accepts 1..32. Increasing worker count does not guarantee a throughput gain; the optimal value depends on browser scheduling, workload size, CPU topology and competing system load. Scalar/worker parity is release-tested across objectives and P values.

22.1 Worker correctness boundary

Workers may reorder independent scoring work, but they cannot change the acceptance rule. The canonical coordinator commits in deterministic order after exact objective comparison. A worker failure is treated as an execution fault, not as permission to commit an unverifiable result.

23. WebGPU architecture

WebGPU is the primary high-performance training path when available. It uses integer path arithmetic and GPU reductions while preserving host ownership of the canonical model/checkpoint state. Browser-exposed adapter identity may be incomplete or withheld; implementation selection therefore relies on fixed compatible kernels, resource preflight, verification and measured behavior rather than on a user-supplied GPU name.

23.1 WebGPU implementations

Implementation	Registry ID	Role
Auto - recommended	auto	Selects the fastest compatible verified implementation for the current workload using uncontended measured timing.
Portable I32 / Chain v1	portable-i32-v1	Correctness baseline: exact i32 chain scoring and u32 class reduction.
Portable I32 / Batched v2	portable-i32-batched-v2	Adds GPU-side objective/winner reduction with compact readback; oriented toward larger candidate/sample jobs.
GPU-resident Cache v3	resident-cache-v3	Keeps Training inputs, topology/parameters and activation cache resident on device and commits affected paths transactionally.

23.2 Verification and runtime protection

A WebGPU implementation cannot train the canonical model until its bounded self-test matches the scalar reference. v1.0.2 verification includes a small fixture and a P=8192 boundary fixture. Optimized implementations can additionally use Runtime shadow verification so selected GPU work is cross-checked against reference results.

Resource preflight estimates buffer sizes, bindings and dispatch requirements against granted device limits before training. A resource-incompatible selected implementation produces a blocking Advisor rule. Device loss triggers one automatic reacquire/reverify attempt before manual Retry GPU / CPU fallback behavior.

Canonical safety boundary
GPU caches are disposable acceleration structures. The host-owned model, dataset, RNG and checkpoint state remain authoritative. An unverified or invalid GPU result cannot be committed to the canonical model.

23.3 Compact winner and Resident Cache v3

Batched v2 and Resident v3 can reduce candidate results on the GPU and return a compact winner rather than reading back all candidate totals. The hardened winner path uses a 16-byte packet containing raw 32-bit words, candidate information, objective low/high words and a candidate-count sentinel. Packet validity is checked before commit.

Resident Cache v3 mirrors Training data, topology/parameters and Training activations on-device. Host/device revision handshakes and transactional commits keep the resident mirror synchronized. After checkpoint restore or device loss, derived resident state is rebuilt from canonical host state rather than serialized.

The finalized implementation includes usage-aware GPU buffer allocation, COPY_SRC support for compact readback, scoped WebGPU validation/OOM handling, and a corrected three-Int32 accepted-node mirror upload.

24. GPU capabilities dialog



Figure 11. GPU capabilities and IGL implementation dialog after successful Resident Cache v3 verification/training.

Control	Options / range	Function and operational meaning
Implementation	Auto; Chain v1; Batched v2; Resident Cache v3	Chooses a WebGPU implementation from the fixed registry.
Verify implementation	Button	Runs exact scalar/GPU parity fixtures and records verification state.
Retry GPU	Button	Reacquires/reinitializes WebGPU after a problem or device loss.
Physical GPU label	Optional text	User/system-confirmed descriptive label only; never sufficient for kernel selection.
Runtime shadow verification	Checkbox; default ON	Cross-checks optimized GPU work against reference behavior.
Copy GPU report	Button	Copies the capability/verification/telemetry/problem report.
Save GPU report JSON	Button	Downloads the GPU report in machine-readable form.
Auto-tune current workload	Button	Measures compatible verified implementations for the active workload; contended timing is rejected for Auto selection.
Open full diagnostics	Button	Opens independent local numerical/backend diagnostics.

24.1 GPU capability and telemetry fields

The dialog presents WebGPU adapter/device information, granted resource limits, features, WebGL2 information, implementation status, preflight and the latest telemetry. Browsers can legally withhold vendor, architecture, device and description strings.

Telemetry / status	Interpretation
prepareMs / uploadMs / encodeMs	Host preparation, transfer and command encoding components where reported.
gpuPathMs / gpuReductionMs	GPU path and reduction timing when timestamp-query support is available.
completionAndReadbackWaitMs	Observed wait for completion/mapping/readback; can expose external contention.
candidateSamples / candidateSamplesPerSecond	Work and throughput measures based on candidate x Training sample scoring.
compactReadbackBytes	Compact winner readback footprint; Resident/Batched optimized path uses 16 bytes.
cacheState	Resident cache validity state such as VALID.
Preflight compatible	Required resources fit within granted limits.

24.2 WebGL2 vs. WebGPU

WebGPU is the compute interface used by accelerated training. WebGL2 is used for network visualization support and is reported separately in the GPU dialog. A WebGL renderer string may be generalized, virtualized or obfuscated and should not be treated as authoritative physical-hardware identification.

Part VI - Benchmarking, diagnostics, persistence and export

25. Benchmark profiles

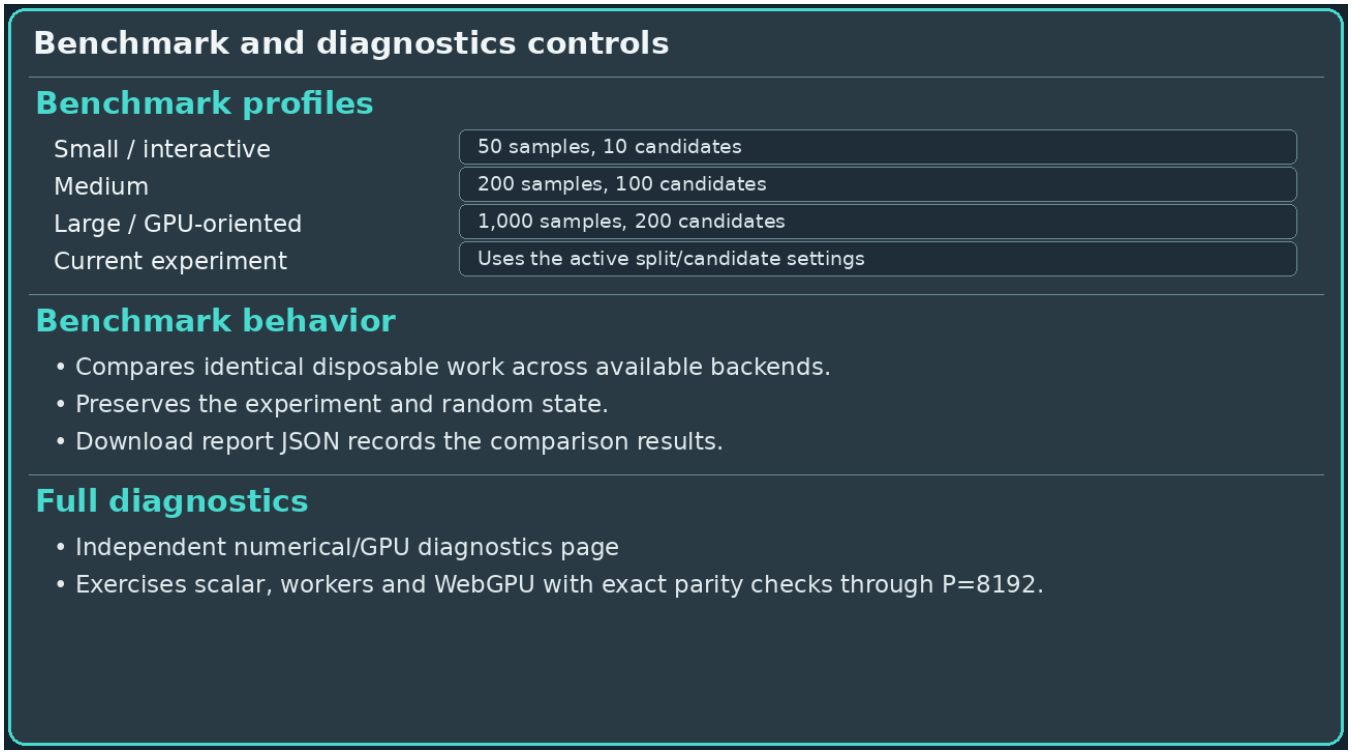


Figure 12. Benchmark profiles and diagnostics reference.

Profile	Samples	Candidates	Intended scale
Small / interactive	50	10	Expose fixed overhead and interactive behavior.
Medium	200	100	Moderate comparison workload.
Large / GPU-oriented	1000	200	Large workload intended to reveal GPU advantages.
Current experiment	Active	Active	Uses current Training/candidate settings.

Benchmark work is disposable: it compares identical backend work and preserves the model, random state and experiment history. Throughput crossover is hardware/workload dependent, so benchmark/Auto-tune evidence is preferred over a static backend prescription.

26. Full diagnostics

diagnostics.html is an independent local diagnostics surface. It exercises exact arithmetic, workers and WebGPU when available. The v1.0.2 diagnostics use the complete P registry through 8192 and include optimized compact-winner selection verification.

Diagnostic JSON is appropriate for deployment acceptance and troubleshooting because it records implementation checks, fixture dimensions, capabilities, limits, telemetry and any failure message without requiring inspection of the normal training UI.

27. Checkpoints and exact resume

A checkpoint preserves the canonical experiment: dataset identity and data, model topology and parameters, scale P, RNG state, training cursor/order for a partial pass, relevant history and disclosure records. Derived CPU/GPU caches are rebuilt.

Checkpoint imports are size-bounded and structurally validated. Dataset hashes, dimensions, glyph provenance, topology, parameter ranges, RNG/pass state and history shapes are checked before the live experiment is replaced. Failed imports are transactional and do not partially mutate the established experiment.

v1.0.0 and v1.0.1 checkpoints remain supported by v1.0.2.

28. Portable C export

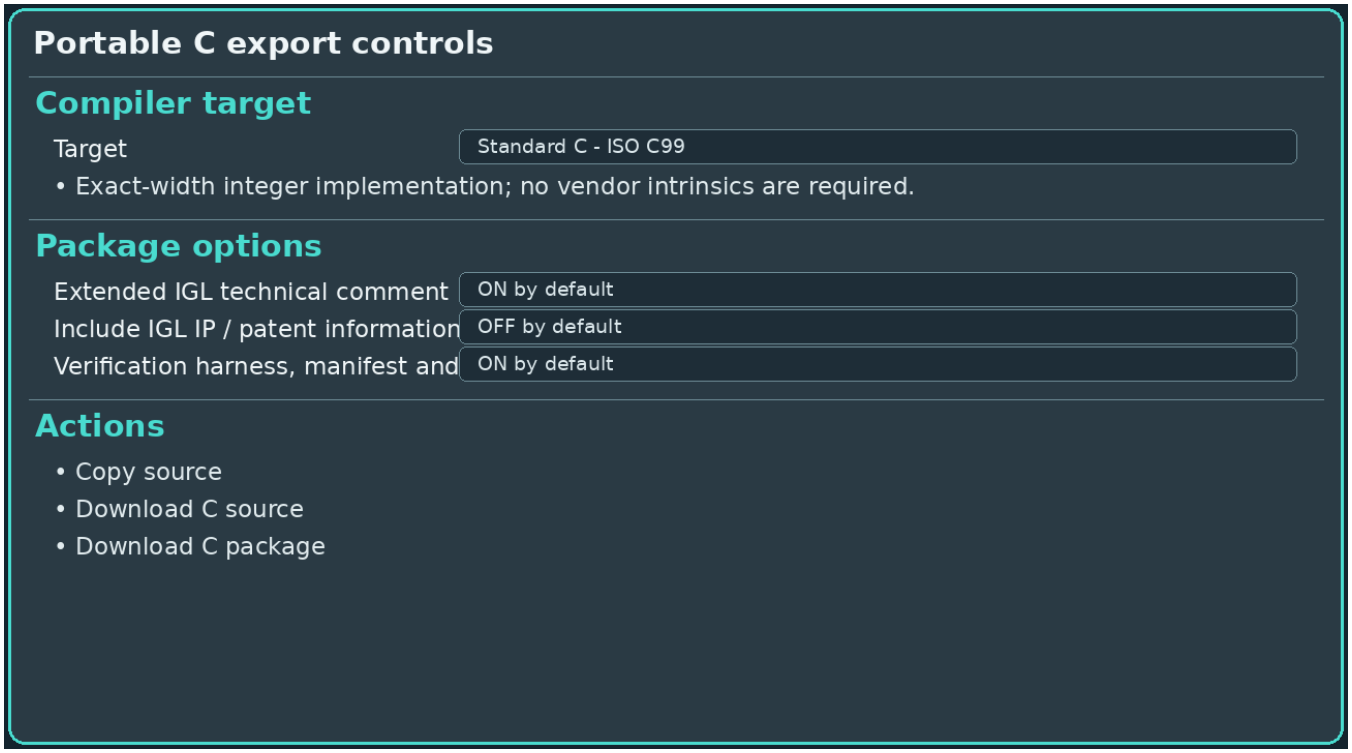


Figure 13. Portable C export controls and package options.

Control	Options / range	Function and operational meaning
Compiler target	Standard C - ISO C99	Portable exact-width integer implementation; no vendor intrinsics.
Extended IGL technical comment block	Checkbox; default ON	Documents IGL, model dimensions/P/tiling, generated functions and exact numerical contract.
IP / patent information	Checkbox; default OFF	Adds informational MLiglon/patent text; explicitly does not grant a license.
Verification harness / manifest / golden vectors	Checkbox; default ON	Adds a small verifier, manifest and golden vectors for target-toolchain parity testing.
Copy source	Button	Copies generated C source.
Download C source	Button	Downloads source text.
Download C package	Button	Downloads the configured multi-file portable inference package.

28.1 C numerical and memory contract

The C API accepts the original W x H quantized input array in row-major order. Replication/tiling and odd-edge zero injection occur inside the generated model. The raw output remains Y in 0..P and classification uses the same midpoint rule as the browser.

Inference is caller-owned and no-heap: the caller provides the documented workspace. Operands are explicitly widened to int32_t before multiplication so behavior does not depend on a target C implementation having a 32-bit native int. The automated suite compiles generated Standard C under strict C99 warnings and undefined-behavior sanitization through the P=8192 Int16 boundary.

29. Metrics and experiment exports

Export	Contents / role
Metrics CSV	Machine-readable current metrics for analysis and archival.
Experiment JSON	Experiment manifest/state identity and reporting metadata.
Dashboard PNG	Image capture of the current monitoring/dashboard state.
GPU report JSON	Capabilities, verification state, resource limits, telemetry and GPU problems.
Benchmark report JSON	Backend comparison results for the selected benchmark profile.

30. Display recovery controls

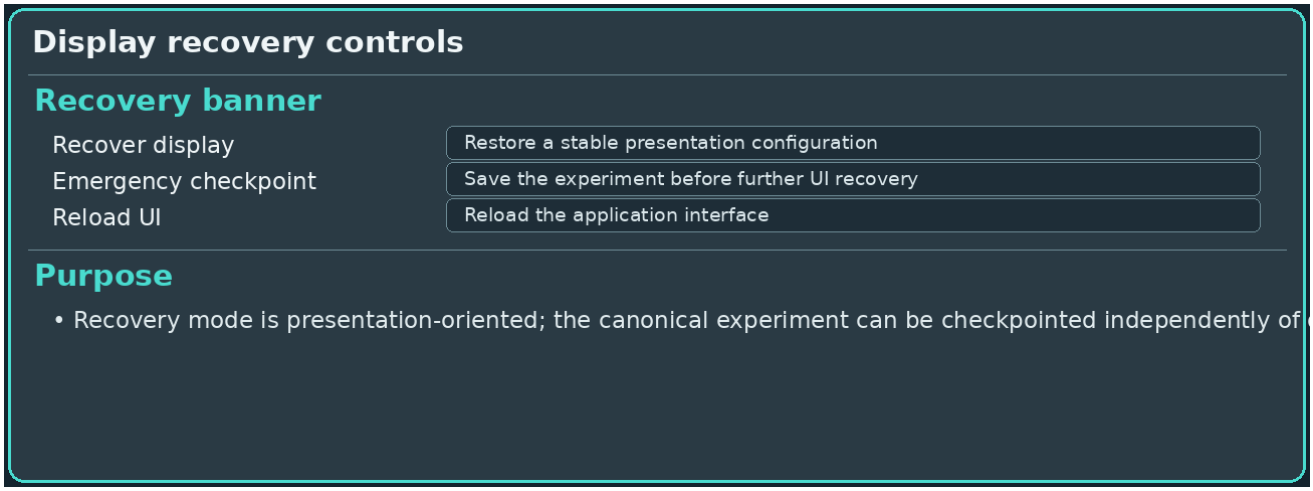


Figure 14. Display recovery banner controls.

If presentation state becomes unusable, the recovery banner can restore display settings, save an emergency checkpoint, or reload the UI. Recovery is separated from canonical experiment state so the experiment can be preserved before presentation recovery.

31. Browser, security and deployment boundaries

IGLnet is a static browser application. Runtime application code contains no fetch, XMLHttpRequest, WebSocket, EventSource or beacon path. The recommended Content Security Policy uses connect-src none. Training/model data remain in the browser unless the user explicitly downloads an artifact.

Workers are same-origin modules. WebGPU kernels operate only on buffers created by the application and are selected from a fixed registry; users cannot supply arbitrary WGSL. Generated C is downloaded as text and is never executed by the web server.

Boundary	Expectation
Runtime network	No application runtime network dependency; self-only scripts/styles/workers.
Checkpoint import	Size-bounded, structurally validated, transactional.
GPU	Fixed implementation registry plus scalar-reference verification.
Fonts	Presentation names use installed fonts/fallbacks; no runtime web-font download.
Server	No deployable .htaccess in the release; web-server security policy belongs in administrator-controlled configuration.

Part VII - Exhaustive glossary

The glossary defines numerical, training, data, visualization, Advisor, CPU/GPU, persistence and export terminology used by the v1.0.2 program and this manual.

16-gate key

Visualization key for Boolean corner patterns and gate names.

Accepted change

A candidate parameter triple committed because it strictly lowers the exact current Training objective.

Accuracy

Fraction of samples classified correctly using the binary threshold 2Y >= P.

Activation region

One of the five piecewise regions that maps S to bounded output Y in 0..P.

Advisor

The continuously evaluated Configuration Advisor that derives facts and matches them against the versioned rule database.

Advisor Apply action

An explicit safe configuration change exposed by selected Advisor recommendations, such as switching to Layer-balanced selection.

Advisor blocking flag

Independent boolean on a rule indicating whether training must be blocked; it is not equivalent to severity color.

Advisor evidence

The explanatory facts attached to a matched rule describing why it applies.

Advisor severity

White, green, yellow or red presentation classification for a rule.

Affected path

The selected computed node and its ancestor chain to the root whose outputs can change when that node parameters change.

Auto GPU implementation

GPU registry option that chooses among compatible verified implementations using workload size and uncontended measured timing.

Auto node selection

Node-selection strategy that uses Uniform for 1x1/2x2 and Layer-balanced for 4x4 tiling.

Auto-tune

GPU dialog operation that benchmarks compatible verified implementations for the current workload and rejects unreliable contended timing.

B

Integer node bias parameter in the legal range -P..3P.

Balanced accuracy

Mean of target-class and non-target-class recall.

Batched v2

Portable I32 / Batched v2 WebGPU implementation with GPU-side objective/winner reduction and compact readback.

Benchmark profile

Named disposable workload definition used to compare compute backends without mutating the experiment.

Binary Cross Entropy (BCE)

Versioned fixed-point lookup-table objective with epsilon 1e-6 and table scale 512.

Binary target

The selected source glyph designated class 1; all other selected glyphs are class 0.

Boolean corner

One of input corner states 00, 01, 10, 11 used to characterize gate behavior.

Candidate

A proposed changed W1/W2/B parameter triple for one visited node.

Candidate domain

Finite set of changed legal W1/W2/B triples at a scale P; size (4P+1)^3-1.

Candidate duplicate

A proposal that duplicates an already-scored triple or rounds to a no-op and therefore does not add unique exploration.

Candidate shortfall

Condition in which bounded proposal draws cannot produce the requested number of unique changed triples.

Candidate-sample

One candidate evaluated on one Training image along its affected path.

Canonical checkpoint

Downloaded representation of experiment state sufficient for validated exact restoration.

Canonical model

Authoritative host-owned model state that accelerated caches mirror but do not replace.

Chain v1

Portable I32 / Chain v1 WebGPU correctness-baseline implementation.

Class 0

Non-target class consisting of selected source glyphs other than the target.

Class 1

Target class consisting of the selected target glyph.

Class balanced weighting

Objective aggregation that gives target and non-target classes equal total influence despite different sample counts.

Clean template

Generated source image with no noise, rotation or translation augmentation.

Compact winner

GPU-reduced best-candidate result returned in a small packet instead of full candidate totals.

Compact winner sentinel

Candidate-count value in the 16-byte compact packet used to detect stale or mismatched readback.

Computed layer

A reduction layer composed of independently parameterized IGL nodes rather than raw input pixels.

Configuration Advisor

Full name of the rule-based live guidance subsystem.

Contours

Optional bands over a heat-map visualization; presentation only.

CPU worker pool

Persistent same-origin worker backend that partitions candidate/sample scoring and reduces exact totals.

D

Widened node dot product D=W1*X1+W2*X2.

Dashboard PNG

Image export of the current dashboard/monitoring state.

Dataset identity

Hash/provenance/dimension information identifying the frozen generated dataset and installed source asset version.

Depth-resolution margin

Advisor fact log2(P)-maximum node-to-output path depth; negative values indicate increasing risk that fine early-layer changes vanish through rounding.

Device loss

WebGPU device becoming unavailable; IGLnet can attempt reacquisition/reverification and rebuild disposable caches.

Display throttle

Maximum speed, Balanced or Detailed presentation update cadence; numerical training semantics are unchanged.

Equal samples weighting

Objective weighting in which every sample has equal aggregate contribution, allowing class imbalance to influence totals.

Exact objective

Integer quantity used for stable candidate comparison and strict adoption.

Experiment JSON

Exported experiment manifest/report artifact.

Experiment monitor

Right-side panel showing metrics, current/best revision state, history and response distributions.

False negative (FN)

Target sample classified as non-target.

False positive (FP)

Non-target sample classified as target.

Final Test

Sealed generated split disclosed only by explicit action; reuse is recorded.

Final Test reuse

A second or later disclosure of the same Test identity, flagged because it becomes exploratory rather than a pristine holdout.

Fit network

View command that scales the complete network into the available viewport.

Focal BCE

BCE-derived objective modulated to emphasize difficult examples; v1.0.2 provides gamma=1 and gamma=2 tables.

Global %

Per-proposal probability of using a global rather than local proposal.

Global proposal

Candidate sampled from the full legal parameter range.

Glyph provenance

Source-set ID, version and SHA-256 information included in dataset identity and checkpoint validation.

Glyph set

Versioned installed fixed-bitmap source family: Alphanumeric 5x7, UTF Symbols 6x8 or Chess Pieces 6x8.

GPU preflight

Resource estimation against the actual granted WebGPU buffer, binding and dispatch limits before execution.

GPU report

Capability, implementation, verification, preflight, telemetry and problem record available for copy/download.

GPU shadow verification

Runtime cross-check used by optimized GPU implementations to protect canonical correctness.

GPU-resident Cache v3

Optimized implementation that keeps Training data, topology/parameters and Training activation cache resident on device.

Heat-map palette

Presentation-only mapping from response values to colors in Sample output heat map mode.

Holdout

Dataset split not used to fit model parameters; Final Test is the explicitly sealed holdout in the normal interface.

Host-owned state

Canonical dataset/model/RNG/history stored in normal application memory rather than GPU cache.

IGL

Integer Gate Logic, the exact-integer node model used by IGLnet.

IGLnet

MLiglon browser laboratory for deterministic IGL model generation, training, inspection, benchmarking and export.

Image size

Generated dataset canvas dimensions: 8x8, 10x10, 16x16 or 32x32.

Input slot

Whether an ancestor receives the affected child in X1 or X2 when a candidate path is evaluated.

Int16 parameter contract

Current storage contract allowing P through 8192 while W and B remain within signed 16-bit range.

Integer scale P

Global response/parameter scale controlling quantization resolution and practical trainable depth.

Layer-balanced selection

Selection strategy that distributes a fixed selected-node budget across layers when the budget permits broad coverage.

Linear Margin

Default threshold-oriented exact integer loss with 0.45P/0.55P zero-loss margins.

Local proposal

Candidate produced by integer-rounded Gaussian perturbation of the current W1/W2/B triple.

Local sigma / P %

Configured local standard deviation as a percentage of P.

Local sigma absolute

Advisor fact P x local sigma fraction, measured in integer parameter steps.

Loss registry

Versioned list of Linear Margin, BCE, Focal BCE gamma=1 and Focal BCE gamma=2.

Loss segment

History interval using one objective definition; chart markers prevent misinterpretation across loss changes.

Majority baseline

Accuracy obtained by always predicting the majority class.

Maximum path depth

Advisor depth fact max(0, layerCount-1).

Metrics CSV

CSV export of experiment metrics.

Model revision

Tracked revision identity for the canonical model; Restore best creates a new revision.

Node

Computed two-input IGL element with independent W1, W2 and B.

Node coloring

Network presentation mode: Gate behavior or Sample output heat map.

Node Inspector

Textual detail panel for selected node parameters, sample inputs, D, S, Y, region and path.

Node selection

Policy selecting which computed nodes are visited in a pass: Auto, Uniform or Layer-balanced.

Nodes %

Unique computed-node fraction selected per pass, rounded upward.

Noise %

Dataset Gaussian noise intensity control with sigma=10.2 x percent before clipping.

Objective segment change

Recorded change in active loss function; values across segments are not directly comparable.

Ordered commit

Deterministic application of accepted node updates in pass order even when scoring is parallel.

P resolution bits

Advisor fact log2(P).

Palette contours

Optional visualization bands applied to heat-map colors only.

Parameter digest

Hash/identity of evaluated model parameters recorded with Final Test disclosure.

Pass

One scheduled visit sequence over a selected subset of computed nodes.

Pass cursor

Checkpointed position within a partially completed pass.

Pass direction

Output-first, random or input-first ordering; cycles across passes.

Physical GPU label

Optional human-confirmed descriptive string in the GPU dialog; informational only.

Portable C

Generated ISO C99 inference implementation of the same integer node contract.

Precision

TP/(TP+FP) for target predictions when defined.

Preflight compatible

GPU resource estimate fits the granted device limits for the selected implementation/workload.

Proposal draw

One attempted global or local candidate generation operation.

Quantization

Mapping continuous-like conceptual changes into the discrete integer state space controlled by P.

Recall

TP/(TP+FN) for the target class.

Repeated templates

Condition where augmentation is disabled and Training/Validation repeat the same fixed source glyph templates.

Resident cache

GPU-resident mirror of Training data/topology/activations used by Cache v3; disposable and rebuildable.

Response Y

Final bounded node/network output in 0..P.

Restore best

Operation restoring earliest best Validation checkpoint as a new revision.

RNG state

Deterministic random-generator state preserved by checkpoints and exact resume.

Rotation +/- degrees

Dataset augmentation control for integer-degree rotation, geometry-limited to avoid clipping.

S

Rounded normalized dot plus bias before activation.

Sample output heat map

Network coloring mode that maps selected-sample node responses to a palette.

Scalar CPU

Canonical single-thread reference backend.

Sealed split

Final Test images hidden in the normal interface until explicit disclosure.

Search budget

Advisor workload estimate based on passes, selected nodes, candidates and Training samples before path-length cost.

Seed

32-bit dataset/experiment random seed.

Selected-node budget

Rounded count of nodes selected from Nodes % for one pass.

Shift X / Shift Y

Dataset integer-pixel translation augmentation controls.

Source glyph

Fixed project bitmap that can be included in generated data.

Strict improvement

Requirement that a candidate objective be strictly lower than the current objective before commit.

Target

Selected positive source glyph/class 1.

Telemetry

Measured backend stage timing, throughput, readback and cache/preflight information.

Theme

Presentation registry choice; numerical state is unchanged.

Tiling

1x1, 2x2 or 4x4 source replication before reduction into the computed network.

Timestamp query

Optional WebGPU feature used for GPU-stage timing when advertised by the adapter.

Topology

Network arrangement determined by source dimensions, tiling and alternating reduction to one root.

Training objective

Exact current objective over the complete Training split.

Training split

Frozen generated split used to score every candidate and define the adoption objective.

True negative (TN)

Non-target sample correctly classified as non-target.

True positive (TP)

Target sample correctly classified as target.

Uniform selection

Legacy strategy that selects computed nodes uniformly from the entire topology.

Validation checkpoint

Saved model state associated with a Validation result and used for Restore best.

Validation split

Frozen generated split used for monitoring/model selection rather than candidate adoption.

W1 / W2

Integer node weights in the legal range -2P..2P.

WebGL2

Graphics API used to support network visualization; reported separately from WebGPU compute.

WebGPU

Browser GPU compute API used by verified IGLnet training implementations.

Weighting

Class-balanced or equal-samples aggregation policy for the active objective.

Worker count

Requested persistent CPU worker-pool size from 1 to 32.

Y/P

Normalized bounded response; explicitly not a calibrated probability.

Appendix A - All current Training Advisor messages

This appendix reproduces the current user-visible rule database from the finalized v1.0.2 advisor-rules.json. Trigger expressions use the shipped fact names and operators so the conditions remain exact. Evidence, consequence and recommendations are reproduced from the data. Severity and blocking are shown separately.

The rule database version is 1.0.2 and contains 33 rules.

Configuration not yet assessable	ID: experiment-unavailable | severity: WHITE | blocking: NO | priority: 5
Trigger	(hasDataset = False OR hasModel = False)
Evidence	Generate a dataset and initialize a model before training configuration can be assessed.
Consequence	Training-specific advice is unavailable until the experiment exists.
Recommendation(s)	Create the dataset and model.

Training split has no target examples	ID: missing-target-class | severity: RED | blocking: YES | priority: 100
Trigger	(hasModel = True AND trainTargetCount = 0)
Evidence	Training target count is zero.
Consequence	The binary objective cannot teach the target class.
Recommendation(s)	Regenerate a dataset containing the target glyph.

Training split has no non-target examples	ID: missing-other-class | severity: RED | blocking: YES | priority: 100
Trigger	(hasModel = True AND trainOtherCount = 0)
Evidence	Training non-target count is zero.
Consequence	The binary objective cannot discriminate the target from other glyphs.
Recommendation(s)	Select at least one non-target glyph and regenerate the dataset.

Severe upper-layer coverage risk	ID: 4x4-uniform-low-coverage | severity: RED | blocking: NO | priority: 95
Trigger	(replication = 4 AND selectionStrategyEffective = uniform AND selectionPercent <= 5)
Evidence	4×4 tiling with uniform low-percent node selection can leave decision-side layers untouched for multiple passes. The root-node selection probability under uniform sampling equals approximately the selected-node fraction.
Consequence	The search may spend substantial work in early layers while the output-side decision path remains unchanged.
Recommendation(s)	Use Auto or Layer-balanced node selection. [action: use-layer-balanced]; Alternatively increase Nodes % substantially.

Upper-layer coverage is weak	ID: upper-layer-coverage-low | severity: YELLOW | blocking: NO | priority: 90
Trigger	(hasModel = True AND minUpperLayerVisitProbability < 0.35 AND selectionStrategyEffective = uniform)
Evidence	One or more upper layers have a low probability of receiving any selected node in a pass.
Consequence	Training can appear stalled even when candidate generation is healthy.
Recommendation(s)	Use Layer-balanced selection or increase Nodes %.

Previous pass skipped decision-side layers	ID: last-pass-upper-zero | severity: RED | blocking: NO | priority: 96
Trigger	(lastPassAccepted = 0 AND lastPassUpperLayersVisited = 0 AND trainingObjectiveZero = False)
Evidence	No upper-layer node was visited in the last pass and no parameter change was accepted.
Consequence	The pass could not directly refine the highest decision layers.
Recommendation(s)	Switch to Layer-balanced selection. [action: use-layer-balanced]

Training and Validation repeat fixed templates	ID: repeated-templates | severity: YELLOW | blocking: NO | priority: 55
Trigger	repeatedTemplates = True
Evidence	Noise, rotation and translation are all disabled; the splits repeat the same fixed source glyph templates.
Consequence	Identical aggregate Training/Validation metrics are expected and do not demonstrate generalization to unseen appearances.
Recommendation(s)	Enable noise, rotation or translation for a generalization experiment.

Few target examples	ID: target-count-low | severity: YELLOW | blocking: NO | priority: 50
Trigger	(hasDataset = True AND trainTargetCount > 0 AND trainTargetCount < 5)
Evidence	The Training split contains fewer than five positive examples.
Consequence	Validation and optimization can be statistically fragile.
Recommendation(s)	Increase the Training split or reduce the source-glyph pool.

Severe noise stress configuration	ID: noise-severe | severity: YELLOW | blocking: NO | priority: 55
Trigger	noisePercent >= 75
Evidence	Gaussian noise standard deviation is at least 765 grayscale levels before clipping.
Consequence	Many pixels will clamp to 0 or 255 and source shape information may be heavily obscured.
Recommendation(s)	Use this range deliberately as a stress experiment.

Requested augmentation is geometry-limited	ID: augmentation-reduced | severity: YELLOW | blocking: NO | priority: 50
Trigger	augmentationReduced = True
Evidence	The safe glyph envelope reduced the requested rotation or translation range.
Consequence	The actual augmentation is weaker than the nominal control request.
Recommendation(s)	Use a larger image canvas or reduce transformation requests.

Deep-layer propagation resolution is severely limited	ID: depth-p-propagation-severe | severity: RED | blocking: NO | priority: 91
Trigger	(hasModel = True AND propagationDepthMarginBits <= -2)
Evidence	Depth/P stress testing shows deepest-layer candidate effects are frequently quantized away before reaching the output when log2(P) trails maximum path depth by two or more bits.
Consequence	Lower layers may contribute little to local refinement even though upper layers and sufficiently large global proposals can still improve the model.
Recommendation(s)	Increase P within the supported range or reduce topology depth.; Retain enough global proposals to avoid relying entirely on fine local changes in deep layers.

Deep-layer propagation resolution is marginal	ID: depth-p-propagation-marginal | severity: YELLOW | blocking: NO | priority: 69
Trigger	(hasModel = True AND propagationDepthMarginBits > -2 AND propagationDepthMarginBits <= 0)
Evidence	P provides only about the same number of binary resolution bits as the deepest node-to-output path consumes under the initialized gain distribution.
Consequence	Fine changes originating in the earliest layers may disappear through integer rounding before reaching the decision output.
Recommendation(s)	If lower-layer training is unproductive, compare with a larger P or a shallower topology.

Local proposal scale is below half an integer step	ID: local-proposal-quantized | severity: YELLOW | blocking: NO | priority: 66
Trigger	(hasModel = True AND localSigmaAbsolute < 0.5 AND localProposalFraction >= 0.25)
Evidence	The configured local standard deviation P×(σ/P) is below 0.5 integer parameter steps while local proposals make up at least one quarter of the search.
Consequence	Many Gaussian local draws round back to the current triple and depend on one-step fallback mutations, reducing local search diversity.
Recommendation(s)	Increase local σ/P, increase the global proposal percentage, or use a larger P.

Requested candidate count exceeds the parameter domain	ID: candidate-domain-insufficient | severity: YELLOW | blocking: NO | priority: 78
Trigger	candidateDomainInsufficient = True
Evidence	The requested candidate count exceeds the total number of distinct changed W1/W2/B triples available at this P.
Consequence	Every visited node must receive fewer unique candidates than requested; additional draws cannot overcome the finite domain.
Recommendation(s)	Reduce Candidates or increase P.

Requested training search budget is large	ID: search-budget-large | severity: YELLOW | blocking: NO | priority: 52
Trigger	(estimatedCandidateSamplesRun >= 100000000 AND estimatedCandidateSamplesRun < 1000000000)
Evidence	The requested passes, selected nodes, candidates and Training samples imply at least 100 million candidate-sample evaluations for this run.
Consequence	Runtime may be substantial and backend/dispatch overhead becomes an important part of the experiment design.
Recommendation(s)	Run the local benchmark or GPU Auto-tune before committing to the full run.; Reduce Passes, Nodes %, or Candidates if the search budget is not intentional.

Requested training search budget is extremely large	ID: search-budget-extreme | severity: RED | blocking: NO | priority: 72
Trigger	estimatedCandidateSamplesRun >= 1000000000
Evidence	The requested run implies at least one billion candidate-sample evaluations before accounting for the length of each affected node-to-output path.
Consequence	A long-running experiment is likely even on an optimized backend; an accidental setting can consume substantial compute without improving search quality.
Recommendation(s)	Benchmark the current workload first and verify that the large pass/node/candidate budget is deliberate.

Very coarse integer scale	ID: coarse-p | severity: YELLOW | blocking: NO | priority: 45
Trigger	(hasModel = True AND P <= 4)
Evidence	P≤4 strongly restricts input, parameter and output states.
Consequence	Local proposals can collapse to duplicates/no-ops and model expressiveness may be constrained.
Recommendation(s)	If training stalls, compare with a larger P.

Candidate generation is degenerating	ID: candidate-duplicate-high | severity: YELLOW | blocking: NO | priority: 65
Trigger	candidateDuplicateRate >= 0.5
Evidence	At least half of recent proposal draws were duplicates or rounded no-ops.
Consequence	Increasing candidate requests may not increase actual exploration.
Recommendation(s)	Increase local σ/P, increase global %, or use a larger P.

Candidate shortfall detected	ID: candidate-shortfall | severity: YELLOW | blocking: NO | priority: 70
Trigger	candidateShortfall > 0
Evidence	The requested number of unique changed parameter triples could not be produced within the bounded draw budget.
Consequence	The node received fewer candidate evaluations than requested.
Recommendation(s)	Inspect P, local σ/P and global proposal percentage.

Candidate exploration is sparse for model size	ID: large-model-candidate-sparse | severity: YELLOW | blocking: NO | priority: 40
Trigger	(nodeCount >= 4000 AND candidateCount <= 2)
Evidence	A large topology is using only one or two candidate parameter triples per visited node.
Consequence	Search breadth may be too low for difficult datasets.
Recommendation(s)	Consider 10–100 candidates per visited node.

Training is progressing before accuracy changes	ID: training-progress-threshold | severity: GREEN | blocking: NO | priority: 20
Trigger	(recentTrainingLossImproved = True AND recentAccuracyUnchanged = True AND trainingObjectiveZero = False)
Evidence	The continuous objective decreased while classification accuracy remained unchanged.
Consequence	Target outputs may be moving toward the threshold without crossing it yet.
Recommendation(s)	Continue training while Validation remains healthy.

Possible overfitting	ID: possible-overfit | severity: YELLOW | blocking: NO | priority: 75
Trigger	(passesCompleted >= 5 AND validationAboveBestFraction >= 0.15 AND recentTrainingLossImproved = True)
Evidence	Training improved while current Validation loss is materially above the best saved Validation result.
Consequence	Further optimization may be fitting Training-specific variation.
Recommendation(s)	Consider Restore best, stronger augmentation, or stopping.

No improving candidate was accepted	ID: one-pass-no-improvement | severity: YELLOW | blocking: NO | priority: 60
Trigger	(consecutiveNoImprovementPasses >= 1 AND consecutiveNoImprovementPasses < 5 AND trainingObjectiveZero = False)
Evidence	The last complete pass accepted no strictly improving parameter change.
Consequence	This can be a normal search plateau or a sign of poor coverage/proposal settings.
Recommendation(s)	Use Explain stalled training for a ranked diagnosis. [action: explain-stall]

Training has repeatedly stalled	ID: repeated-no-improvement | severity: RED | blocking: NO | priority: 92
Trigger	(consecutiveNoImprovementPasses >= 5 AND trainingObjectiveZero = False)
Evidence	Five or more complete passes accepted no improving change.
Consequence	The current search configuration is unlikely to recover quickly without intervention.
Recommendation(s)	Use Explain stalled training and address the highest-ranked cause. [action: explain-stall]

Topology may have insufficient capacity	ID: possible-low-capacity | severity: YELLOW | blocking: NO | priority: 45
Trigger	(passesCompleted >= 20 AND trainingObjectiveZero = False AND coverageHealthy = True AND candidateHealthy = True AND replication < 4)
Evidence	Training has run many passes with healthy coverage and candidate generation but retains substantial objective error.
Consequence	The current topology may not represent the dataset efficiently enough.
Recommendation(s)	Consider a larger tiling factor after saving the current experiment.

Model capacity is large relative to dataset diversity	ID: possible-excess-capacity | severity: YELLOW | blocking: NO | priority: 35
Trigger	(uniqueTrainImages > 0 AND nodesPerUniqueImage >= 100)
Evidence	The number of trainable nodes greatly exceeds the number of unique Training images.
Consequence	Training may be slow without providing additional generalization value.
Recommendation(s)	Compare with a smaller tiling factor.

Focal loss with equal-sample weighting on an imbalanced split	ID: focal-imbalance-equal | severity: YELLOW | blocking: NO | priority: 45
Trigger	(focalLoss = True AND weighting = equal AND classImbalanceRatio >= 4)
Evidence	Focal modulation does not itself guarantee equal influence from the two classes.
Consequence	The majority class may still dominate aggregate loss.
Recommendation(s)	Consider Class-balanced weighting.

Selected GPU implementation is not verified	ID: gpu-unverified | severity: RED | blocking: YES | priority: 100
Trigger	(backend = webgpu AND gpuVerified = False)
Evidence	WebGPU training is selected but the active implementation has not passed the scalar-reference parity test.
Consequence	GPU training is blocked to protect numerical correctness.
Recommendation(s)	Run Verify implementation or select a verified implementation.

Selected GPU implementation exceeds granted resources	ID: gpu-preflight-failed | severity: RED | blocking: YES | priority: 100
Trigger	(backend = webgpu AND gpuPreflightCompatible = False)
Evidence	GPU resource preflight reports an incompatible buffer, binding or dispatch requirement.
Consequence	The selected implementation cannot safely execute this workload.
Recommendation(s)	Use Auto so IGLnet can choose a compatible fallback.

GPU benchmark appears externally contended	ID: gpu-contended | severity: YELLOW | blocking: NO | priority: 70
Trigger	gpuContended = True
Evidence	GPU completion/map wait dominates measured kernel execution or repeated throughput is highly variable.
Consequence	This timing is unsuitable for updating Auto implementation selection.
Recommendation(s)	Repeat performance benchmarks when competing GPU workloads are idle.

Final Test has already been disclosed	ID: final-test-reused | severity: YELLOW | blocking: NO | priority: 65
Trigger	finalTestReused = True
Evidence	This Final Test split has already been viewed during the experiment history.
Consequence	Subsequent Test measurements should be treated as exploratory/reused holdout measurements.
Recommendation(s)	Avoid using repeated Final Test results to tune the model.

Layer-balanced budget cannot cover every layer	ID: layer-balanced-budget-insufficient | severity: YELLOW | blocking: NO | priority: 88
Trigger	(selectionStrategyEffective = layer-balanced AND layerBalancedGuaranteed = False)
Evidence	The selected-node budget is smaller than the number of computed layers, so even Layer-balanced selection cannot visit every layer in one pass.
Consequence	Some layers must be skipped despite using Layer-balanced selection, which can make a pass appear unexpectedly unproductive.
Recommendation(s)	Increase Nodes % until the selected-node budget is at least the layer count.

Settings appear acceptable	ID: healthy | severity: GREEN | blocking: NO | priority: 1
Trigger	(hasModel = True AND blockingIssueCount = 0)
Evidence	No known blocking configuration problem was detected.
Consequence	Training may proceed; model quality still depends on data, topology and search behavior.
Recommendation(s)	Monitor Training, Validation and advisor history as the experiment evolves.

Appendix A-2 - Explain stalled training diagnosis messages

The following messages are generated by advisor.js in addition to the persistent rule database. Evidence strings can contain live values such as P, layer count, depth margin, duplicate percentage and plateau count.

Severity	Title	Evidence / condition	Recommendation
Red	Upper-layer coverage inadequate	No decision-side node was visited in the latest pass and Training objective is not zero.	Use Layer-balanced selection or increase Nodes %.
Yellow	Candidate generation degraded	Recent duplicate/no-op rate >=50% or candidate shortfall >0.	Increase local sigma/P, global proposals, or P.
Green	Candidate generation healthy	No high duplicate/no-op rate and no shortfall.	Do not change candidate settings first.
Red	Deep-layer propagation resolution is severely limited	Depth-resolution margin <= -2 bits.	Increase P, reduce topology depth, or retain enough global proposals.
Yellow	Deep-layer propagation resolution is marginal	Depth-resolution margin <=0 but >-2.	Compare with larger P or shallower topology if lower layers stop contributing.
Yellow	Quantization is coarse	P<=4 when no stronger depth warning applies.	Compare with a larger P.
Green	Quantization/depth margin appears adequate	No severe/marginal depth warning and P>4.	Do not change P first unless other evidence indicates quantization limits.
Yellow	Local proposal scale is below half an integer step	Absolute local sigma <0.5 and local proposal fraction >=25%.	Increase local sigma/P, global proposals, or P.
Red	Persistent search plateau	Five or more consecutive passes accepted no change.	Address the highest-ranked coverage/candidate issue before adding more passes.
White	Capacity not yet implicated	Fewer than three passes completed.	Do not increase tiling solely because of this stall.

Appendix B - Exhaustive control and option inventory

This compact inventory is intended for auditing and quick lookup. The main chapters provide the functional explanations and figures.

B.1 Header and Appearance

Control	Options / range	Function and operational meaning
Guide	Button	Built-in guide and shortcuts
Appearance	Popover	Theme, typeface, text size, Samples width, Results width, reset widths
GPU	Button/status	GPU capabilities and implementation dialog
Fullscreen	Toggle	Browser fullscreen
Theme	Graphite, Midnight, Electric Cobalt, Amethyst, High Contrast, Phosphor, Slate Medium, Teal Medium, Sepia Medium, Ocean Blue, Evergreen, Warm Sand, Glacier Steel, Ivory & Gold, Rose Quartz, Technical Paper	Presentation theme
Typeface	Noto Sans (if installed), IBM Plex Sans, IBM Plex Serif, IBM Plex Mono, System Sans, System Serif, System Monospace, Arial / Helvetica, Georgia, Noto Serif (if installed), Noto Mono (if installed)	Local font preference/fallback
Text size	12 px, 13 px, 14 px, 16 px, 18 px, 20 px, 23 px, 26 px, 29 px, 32 px, 36 px, 39 px	UI text scale
Samples width	18-30%	Desktop layout
Results width	20-32%	Desktop layout
Reset panel widths	Button	Restore default layout

B.2 Dataset

Control	Options / range	Function and operational meaning
Glyph set	Alphanumeric 5x7; UTF Symbols 6x8; Chess Pieces 6x8	Installed bitmap source
Image size	8x8; 10x10; 16x16; 32x32	Generated canvas
Target	Any selected glyph	Positive class
Train	10; 100; 200; 1000	Training split size
Validation	10; 100; 200; 1000	Validation split size
Final Test	10; 100; 200; 1000	Sealed split size
Seed	0..4,294,967,295	32-bit reproducible seed
New seed	Button	Generate a fresh seed
Noise %	0..100	Frozen Gaussian noise intensity
Rotation +/- degrees	0..15	Geometry-limited rotation
Shift X +/-px	0..32	Horizontal translation
Shift Y +/-px	0..32	Vertical translation
Generate dataset	Button	Create new frozen experiment data
Set defaults	Button	Set source-set default selection
All	Button	Select all glyphs
Clear	Button	Clear glyph pool
Individual glyphs	62 / 36 / 12 buttons	Inclusion in generated source pool

B.3 Model, search and training

Control	Options / range	Function and operational meaning
Scale P	1..8192 powers of two	Global integer scale
Tiling	1x1; 2x2; 4x4	Source replication/topology
Initialize model	Button	Create/recreate model
Compute	WebGPU; CPU workers; Single CPU	Training backend
Workers	1..32	Worker pool size
Display	Maximum speed, Balanced, Detailed	UI refresh throttle
Benchmark	Small / interactive, Medium, Large / GPU-oriented, Current experiment	Disposable workload profile
Benchmark button	Button	Compare backends
Passes	1; 10; 100; 1000	Requested pass count
Nodes %	1..100	Unique nodes selected per pass
Node selection	Auto — recommended, Uniform — legacy v1.0.x, Layer-balanced	Coverage strategy
Candidates	1; 2; 5; 10; 100; 200	Requested unique changed triples
Global %	0..100	Global proposal probability
Local sigma/P %	1..100	Local proposal standard deviation
Weighting	Class balanced; Equal samples	Objective class aggregation
Error function	Linear Margin · 0.45P / 0.55P, Binary Cross Entropy, Focal BCE · γ=1, Focal BCE · γ=2	Loss registry
Always roll up	Button/persistent	Clamp Advisor text/evidence
Hide text	Button/persistent	Hide Advisor body text
Restore text	Button/persistent	Restore normal expansion
Explain stalled training	Button	Ranked Advisor diagnosis
Apply recommendation	Conditional button	Safe explicit Advisor action
Train	Button	Run requested passes
Pause	Button	Pause at commit boundary
Evaluate	Button	Refresh Training/Validation
Final Test	Button	Disclose sealed Test
Restore best	Button	Restore earliest best Validation checkpoint
Show active node	Checkbox	Follow/highlight active node

B.4 Visualization, monitoring and exports

Control	Options / range	Function and operational meaning
Node coloring	Gate behavior; Sample output heat map	Network color mode
Heat-map palette	Thermal, Topographical, Viridis, Plasma, Inferno, Magma, Turbo / Rainbow, Blue–Red diverging, Grayscale, High-contrast discrete	Response palette
Contours	Checkbox	Heat-map contour bands
Zoom	1..30, step .25	Network magnification
Fit	Button	Fit network
16-gate key	Expandable	Gate corner-pattern legend
Log	Checkbox	Log history scale
Response dataset	Training; Validation; Final Test	Response plot split
Save checkpoint	Button	Download checkpoint
Load checkpoint	Button/file	Validated restore
Export C	Button	Open C export
Metrics CSV	Button	Export metrics
Experiment JSON	Button	Export manifest/report
Dashboard PNG	Button	Export dashboard image

B.5 GPU, Benchmark, Export and Recovery dialogs

Control	Options / range	Function and operational meaning
GPU Implementation	Auto — recommended, Portable I32 / Chain v1, Portable I32 / Batched v2, GPU-resident Cache v3	Select fixed GPU implementation
Verify implementation	Button	Run scalar/GPU parity self-test
Retry GPU	Button	Reacquire/reverify
Physical GPU label	Text <=120 chars	Informational label
Runtime shadow verification	Checkbox; default ON	Optimized GPU runtime parity protection
Copy GPU report	Button	Copy report
Save GPU report JSON	Button	Download report
Auto-tune current workload	Button	Measure verified implementations
Open full diagnostics	Button	Open diagnostics.html
Download benchmark report JSON	Button	Save benchmark results
Compiler target	Standard C - ISO C99	C target registry
Extended comments	Checkbox; ON	Technical comments
IP/patent comments	Checkbox; OFF	Informational IP block
Verification harness	Checkbox; ON	Golden vectors/verifier/manifest
Copy source	Button	Copy C
Download C source	Button	Download source
Download C package	Button	Download package
Recover display	Button	Restore presentation
Emergency checkpoint	Button	Checkpoint during recovery
Reload UI	Button	Reload interface

Appendix C - Installed source glyph contents

C.1 Alphanumeric 5x7

62 glyphs: 0-9, A-Z, a-z. Default selection is 0-9 and default target is 0.

C.2 UTF Symbols 6x8

36 glyph labels: Left arrow, Right arrow, Up arrow, Down arrow, Northwest arrow, Northeast arrow, Southeast arrow, Southwest arrow, Open circle, Filled circle, Open square, Filled square, Open triangle up, Filled triangle up, Open triangle down, Filled triangle down, Open diamond, Filled diamond, Plus, Minus, Multiply, Divide, Equals, Not equal, Less than, Greater than, Less than or equal, Greater than or equal, Star, Check mark, Cross mark, Heart, Diamond suit, Club suit, Spade suit, Sun.

Default target is Star; all symbols are selected by default.

C.3 Chess Pieces 6x8

12 grayscale pieces: White King, White Queen, White Rook, White Bishop, White Knight, White Pawn, Black King, Black Queen, Black Rook, Black Bishop, Black Knight, Black Pawn. White pixels use grayscale 255, black-piece pixels use 128, and background uses 0.

Default target is White King; all pieces are selected by default.

Appendix D - v1.0.2 verification summary

The finalized v1.0.2 production tree passed 95/95 automated test groups, repeated unified stress runs, all 168 image-size x tiling x P configurations with a real strict-greedy step, 100,000 randomized legal node evaluations per stress run, exact scalar/worker parity, P=8192 C/UBSan boundaries, checkpoint round-trip, Advisor reachability/fuzzing, deployable-tree security checks and package integrity checks.

The authoritative deployed Firefox/Windows environment additionally passed Resident Cache v3 exact CPU/GPU verification and bounded shadow-verified real training with compactReadbackBytes=16 and cacheState=VALID. Hotfix 8 live acceptance also passed the P=8192 GPU/training boundary, Advisor display controls, multi-row Source Glyph viewport and high-P checkpoint behavior.

Interpretation
Release testing establishes consistency of the shipped implementation and reference backends. It does not claim identical performance on every GPU/CPU/browser or target-MCU timing/energy; those are deployment measurements.

Document end

IGLnet v1.0.2 combines deterministic integer learning, extensive configuration flexibility, robust multi-backend compute, deep inspection and an expanding evidence-based Training Advisor in a single standalone browser laboratory.