The Open Small Models Accord

On open, accessible, and correctable AI for the next decade

Anivar Aravind

Version 0.2 (draft) · Issued 25 April 2026

Contents

The Open Small Models Accord

On open, accessible, and correctable AI for the next decade

Version 0.2 (draft) · Issued 25 April 2026

Preamble

The open systems that run modern computing share a single condition. The substrate has to remain modifiable by the people who depend on it. AI will be built either as inspectable infrastructure or as opaque services, and the choice is being made now.

This document sets out ten principles necessary for small language models, and the systems that run them, to be open in a meaningful sense. The principles are compatible with existing open source standards and with public-benefit technology traditions. Researchers, model developers, infrastructure providers, civil society organizations, enterprises, and standards bodies should be able to endorse them without contradicting the work they already do.

Endorsement signals agreement with these principles and a commitment to move toward them in practice. It is not a claim of completion.

1. Small models are where openness, accessibility, and accountability can actually meet.

Frontier AI systems are now built at a scale that puts full openness beyond the reach of anyone outside a handful of firms. Training a frontier model requires compute, data, and operational resources that no university, civil society organization, or public institution can assemble. Open weights released from these models are useful, but they are not open systems, because the people who receive them cannot reproduce, retrain, or meaningfully audit what they have been given.

Small language models are different. At the scale where a model can run on a phone, a laptop, or a modest server, the full apparatus of open source becomes possible again: reproducible training runs, portable weights, documented data, community fine-tuning, independent evaluation. What is aspirational at the frontier is achievable here.

The comparison to mobile computing is useful. Smartphones changed who got to use computers not by being more powerful than desktops but by reaching people who would never have used one. AI is following the same pattern. Cloud-hosted frontier systems serve the populations already served by cloud infrastructure. Models small enough to run locally reach everyone else.

Openness on its own is not the point. The point is what openness enables: the capacity of the people a system affects to inspect it, contest it, correct it, and refuse it. National or vendor control of AI infrastructure is hollow when the system stays irreversible for those it excludes, misclassifies, or fails. A system is genuinely open not because someone owns its software but because the people it governs can do something about it when it goes wrong. We wrote this accord because small models are where that capacity can actually be built, at the scale where it affects the most lives.

2. Openness is a layered property; disclosure follows the layer.

An AI system is open to the extent that someone other than its original developer can reproduce it, inspect it, and change it. The Open Source Initiative has defined this carefully for AI in its Open Source AI Definition, requiring access to code, parameters, and information about training data. We draw on that work and extend it by adding a fourth layer.

We call the four layers **Logic, Weights, Data, and Representation**, together **LWD-R**. The principle behind the four layers is that anything which shapes the operative behavior of the deployed system falls within disclosure obligations, regardless of where in the system it sits.

Logic is the model architecture, the inference code, the tokenizer, and any deployment-time transformations applied between training and inference. For architectures with internal routing or gating, Logic includes the routing networks alongside the expert components. The tokenizer matters particularly for language coverage: a tokenizer that does not represent a language well constrains what any model built on it can learn about that language.

Weights are the trained parameters, released under terms that allow use, redistribution, and modification. Where weights are deployed in transformed form — quantized, distilled, pruned, or merged with adapters — the transformations are documented sufficiently for independent reproduction. For mixture-of-experts and similar architectures, weights include the gating networks alongside the experts. Reproducibility is assessed against the complete deployed system, not against active parameters alone.

Data is the training corpus and the artifacts that shaped what the model learned from it: filtering and curation procedures, training schedules where they affect outcomes, and routing or gating training data for architectures that use them. Where training data is itself derivative, including synthetic data generated by other models or expert outputs distilled from larger

systems, the provenance extends to the source. Transparency is not transferable through synthetic generation. For models trained through reinforcement learning, the Data layer extends to the reward models, verifiers, exploration policies, and procedural specifications that shaped the training signal.

Representation is the operative categorical schema the deployed system uses to interpret the world. Quantization, distillation, pruning, MoE routing decisions, and the training procedures that shape representational geometry all modify R. R-layer disclosure must describe the deployed system's operative representation, not a nominal representation that no deployment uses. Where the deployed R derives from another model's R through synthetic training data, distillation, or weight inheritance, the provenance extends through that derivation chain. Communities the model classifies should be able to review the operative R and contest it.

Without representation, openness stops at mechanics and never reaches meaning. A system can expose its code and weights while embedding classifications that cannot be examined or contested. A system can publish its trained weights while deploying a quantized derivative whose representational geometry differs from the disclosed artifact. A system can publish its expert weights while keeping its routing logic proprietary. R-layer disclosure makes these failure modes visible.

Not every release satisfies every layer, and the accord does not pretend otherwise. A base model release is assessed for the full training it represents. A derivative — fine-tune, adapter, distillation, domain or task adaptation, or quantized variant — is assessed only for what that derivative adds, with the base model's own disclosure referenced rather than reproduced. Each stage discloses its own work. The openness of the whole system is the composition of what each stage contributes.

Endorsement commits signatories to progress on the layers they touch, to document clearly which layers their releases meet, and to treat full openness across the stack as the long-term goal. Few releases in 2026 meet all four layers fully. The accord asks signatories to make visible where they are, and to move.

3. Open models are built at the scope of a specific domain, task, and language.

Small models work well not because they are miniature versions of large ones but because they serve a different purpose. A useful small model is usually scoped to a specific domain, tuned for a specific task, and built for a specific language or language mix. Agricultural question-answering in Kannada. Legal summarization for Indian tax law. Clinical discharge notes in Bangla. Maintenance log extraction in a particular industrial vocabulary. Each of these is a bounded problem, and small models are the natural architecture for bounded problems.

This inverts the frontier thesis. A frontier model tries to serve every domain, every task, and every language at once, which forces compromises on all three. A small model that serves one domain, one task, and one language mix can actually be good at that specific thing. It can also be owned and maintained by the institution that deploys it, because the institution can fine-tune it on its own data without sending that data to a vendor.

Three consequences follow. Deployment goes to the SLM layer because the organizations doing the deployment need models that understand their specific work. Openness at the SLM layer is a precondition for this, because an institution cannot fine-tune a frontier model on proprietary data without surrendering it to the vendor. Language coverage stops being about training ever-larger models on ever-more-languages and starts being about enabling the communities that need a model in their language to build one. Tokenizer openness is part of language coverage; a community cannot meaningfully build a model in its language using a tokenizer that fragments the language into subwords trained primarily on English.

We commit to supporting this deployment pattern as a first-class way to build AI, and to designing the models, tools, tokenizers, benchmarks, and documentation that make it practical.

4. Running a model is not the same as reproducing it.

Inference is not reproduction. A model that runs on modest hardware is not automatically accessible in any meaningful sense. Mixture-of-experts architectures and other efficiency designs can make a model cheap to run while keeping it expensive to retrain by activating only a subset of parameters per token. Reporting only active parameters obscures this gap. Two capacities need to be distinguished. **Inference forkability** is the ability to take a released model and run it. **Training forkability** is the ability to reproduce or fundamentally alter it. Only the second produces accountability.

Where reproducing a model requires compute well beyond what research institutions, public bodies, enterprises, or independent developers can assemble, openness in principle becomes narrowness in practice. This is **compute capture**: when training cost so far exceeds accessible compute that legal forkability becomes meaningless without practical reproducibility.

Compute capture is not the only barrier. Modern small models are increasingly trained on artifacts produced by frontier models: synthetic instruction data, evaluation sets, reasoning traces, distilled experts, reward models, and verifiers used in reinforcement learning. Where these dependencies exist, the small model inherits ontology from the upstream system, and reproduction requires reproducing the pipeline that produced the training signal. We call this **data capture**: when the data required to train a model can only be produced by infrastructure the open community cannot reproduce. Data capture and compute capture together determine whether training forkability is genuinely available.

We support the development of public, academic, federated, and community-governed compute resources. We support open synthetic data pipelines, openly-trained generator models, and open reward models and verifiers for RL training. We commit to publishing honest training cost estimates, inference cost estimates, total and active parameter counts where architectures distinguish them, data manifests, and training pipeline provenance, so that anyone can assess whether a given model is actually reproducible. Active parameter counts, compute used, and other proxies are not substitutes for disclosed cost.

5. Local inference keeps accountability with the people affected.

The capacity to inspect, contest, correct, and refuse an AI system only remains available when the inference happens somewhere the affected people can reach. Centralized inference concentrates that capacity in whoever operates the servers. Local inference — on the user's device, on community or cooperative infrastructure, or on institutional clusters — distributes it to where it is actually exercised.

A model running on a clinic's own server cannot be remotely disabled, silently updated, or metered by token. A model running on a farmer's phone does not require her data to leave her hand. A model running on a community server in a region with intermittent connectivity continues to work when the connection fails. These are properties the cloud cannot deliver no matter how cheap its API becomes.

Local inference is not automatically free of capture. Mobile devices and consumer hardware run models through proprietary neural processing units, closed runtime layers, and gatekept distribution channels. Edge deployment escapes cloud capture but remains within hardware capture if the hardware substrate is not addressable by the open community. Standardized inference runtimes, portable model formats such as ONNX and GGUF, open execution specifications

where commercially feasible, and distribution pathways that do not depend on closed app store gatekeeping are part of what makes local inference open in practice.

We commit to prioritizing deployment topologies that keep inference local without losing essential function, to building the models, runtimes, and tooling that make local-first deployment a mature option across the hardware that actually exists in the world, and to working toward hardware-layer transparency where it is currently absent.

6. Safety comes from inspectable architecture, not from closure.

Concerns about AI safety are legitimate, and some of them are urgent. The question is what architectural response those concerns should produce. We believe that response is inspectable, reproducible systems with deterministic boundaries around what the model can do, not restricted access to weights, training data, or operational logic.

Closing a system in the name of safety does not remove the safety problem. It relocates accountability from the system's architecture to the vendor's judgment, and it puts the power to decide what is safe in the hands of the same actors whose commercial interests are served by holding that power. The opposite approach produces more accountability: systems whose behavior can be independently verified, whose failure modes can be studied in public, and whose corrections do not depend on the operator's consent.

Safety in this sense is measurable. The dimensions, drawn from the aviation, nuclear, and automotive engineering traditions that have handled safety-critical systems for decades, are consistency under repetition, robustness under perturbation, predictability in how a system fails, and severity bounds on what happens when it does. These measures are independent of raw capability. A more capable system is not automatically a safer one. A scoped system is easier to evaluate than an unbounded one.

7. Agents are systems, and the system is what requires openness.

Agentic AI is not a model that takes actions. It is a system in which a model is one component among several, and the components together produce the behavior the user encounters. Openness applies to the system.

The action boundary is the deterministic policy layer that decides whether a proposed action is allowed. The model proposes; the boundary decides. The boundary is inspectable, machine-readable, and outside the prompt context. The same transparency that helps operators specify what agents may do helps defenders identify when agents are being exploited.

The harness is the architecture that turns a model into something agentic. It manages memory across steps, decomposes plans, selects tools, parses outputs, recovers from errors, and orchestrates flow between model inference and tool execution. A closed harness around an open model produces a system whose behavior cannot be reproduced from the model alone. The harness should be disclosed at the same standard as the model: its architecture, its prompts and templates, its memory management, its tool selection logic, its error handling, and its version history.

Skills and tools are the composition primitives within the harness. We support open standards for skill and tool specification and treat skill specifications, tool definitions, harness implementations, and action boundaries as first-class open artifacts alongside the models they constrain.

Deployment-time variables shape behavior alongside these structural components: system prompts, decoding parameters, retrieval pipelines, and output filters. They belong in the disclosure of the deployed system. A retrieval-augmented system whose retrieval index,

embedding model, chunking strategy, and reranking model are opaque is not an open system regardless of the underlying model's openness.

Agent evaluation needs multi-run protocols, systematic perturbation testing, and the release of transcripts detailed enough for independent reviewers to check whether the system actually behaved as its designers claimed.

8. Evaluation must measure what deployment requires, not what is convenient to report.

A system is reproducible in the open source sense when an independent party, given what has been published, can rebuild it and verify that it behaves as claimed. For AI this requires reproducible training procedures where feasible, clear records of compute and data provenance, and evaluation protocols that measure what actually matters once the system is deployed.

Benchmarks that reduce a system to a single accuracy score hide the differences that determine whether the system is fit for use. A model that succeeds seventy percent of the time does not reliably succeed. A model that is cheaper to run than a competitor may be more expensive to retrain. Useful evaluation reports capability alongside reliability, and accuracy alongside cost, in terms someone else can reproduce. Pareto-frontier reporting, multi-run protocols, transcript release, and honest cost disclosure are what separate evaluation from marketing.

Evaluation harnesses are themselves artifacts that determine outcomes. Two evaluation runs of the same model on the same benchmark using different harness configurations can produce substantially different results. Where evaluation results are reported, the harness used should be specified and reproducible. For capabilities that resist benchmarking, including long-horizon tasks, open-ended agent work, and tasks whose success is qualitative, open-world evaluations with documented human intervention and qualitative log analysis complement benchmarks rather than replacing them.

We commit to publishing our work in forms that support independent verification at the scope of each contribution, including transcripts, multi-run statistics, harness specifications, and cost disclosures. We commit to engaging with standards and regulatory processes as they mature.

9. Coordination through open standards bodies matters now, while the choices remain contestable.

Open infrastructure benefits from coordination through standards bodies including the Open Source Initiative, IETF, W3C, and specific communities within broader foundations where substantive work gets done. We support contributing to these venues when participation is open and the work serves the principles here.

No venue is inherently legitimate. Legitimacy comes from whether the venue can be inspected, challenged, and improved. Participating in a venue is a practical choice about where work happens, not an endorsement of its governance. Where established venues cannot be corrected through participation, federation, alternative venue formation, and evolution of the governance itself are legitimate responses.

The protocols, runtimes, model architectures, and release practices that get normalized in the next few years will shape what is possible for a decade after that. Agent orchestration standards are being consolidated. On-device AI runtimes are being wired into mobile and desktop operating systems. National AI frameworks are being drafted. Releases that meet only some of the layers the accord names are being treated as "open" in policy documents.

These choices are still being made. They will not be in five years. The accord exists because the present moment still rewards coordinated direction.

10. Accountability requires visible disclosure, community contestation, and revisable commitments.

Endorsing this accord is a statement of direction, not a claim of arrival. Signatories will fall short of these principles in specific releases and specific decisions. What they commit to is making those shortfalls visible through public LWD-R disclosure on their own releases, and working toward closing them over time.

Disclosure happens at the release. A signatory publishing a model, a derivative, an adapter, a harness, or a deployment documents which LWD-R layers the release meets and which it does not. The form of disclosure is the signatory's choice: a model card, a release note, a project repository, or any other artifact that makes the work checkable by the people the system affects. What matters is that the disclosure travels with the release and can be referenced by anyone evaluating it.

Where a deployed model produces representational categories that affect specific communities, particularly in administrative decisions, eligibility determinations, or other consequential classifications, the affected communities should be able to review those categories and raise concerns about them. We encourage signatories to specify how they receive and respond to such concerns, and to make their response practices public. The accord does not specify a single mechanism. Different signatories serve different communities and will appropriately use different approaches.

The accord is versioned. It can be revised through open process as the technical, governance, and regulatory ground shifts. Signatories can withdraw. The community can contest endorsements that diverge from corresponding practice. We encourage participating jurisdictions, multilateral frameworks, and procurement bodies to develop their own evaluation processes that draw on the principles here, recognizing the work signatories actually do rather than the fact of endorsement alone.

The Open Small Models Accord, version 0.2 (draft), issued 25 April 2026, authored by Anivar Aravind. Hosted at openslm.ai. Released under CC0 1.0. Translate, adapt, republish. Endorse by signing. Build according to the principles.