Introducing Inkling 2 Mini

Nov 20, 2026

• Capabilities
Architecture and training
Monitorability of looped transformers
Try Inkling 2 Mini today

Try on Tinker Hugging Face > Model card >

Today we are releasing Inkling 2 Mini, a fast, powerful model that's small enough to run on your laptop.

Inkling 2 Mini is a looped transformer, running the middle layers of the model three times for every token. This allows the model to apply more reasoning and reach the right answer with fewer tokens, especially on tasks such as agentic coding that depend on reasoning quality. Inkling 2 Mini leads every open-weights model with less than 100B total parameters on benchmarks such as DeepSWE v1.1 (48.2%) and Vals Index v2.2 (46.5%). It runs faster than similarly performing models as well.

Figure 1: Reported coding scores for models under 100B total parameters. The green line is the frontier with Inkling 2 Mini, the dashed line the frontier without it, and the shading the ground it adds. Decoding speed is a theoretical ceiling at 614 GB/s with BF16 weights and batch size one, including repeated reads across loops. Evaluation setups vary by model.

Inkling 2 Mini is a Mixture-of-Experts model with 26.4B total parameters, 3.4B active, and 5.4B weight activations per token to account for looping. Looping allows the model to reason longer without writing its reasoning down, which matters for monitoring: our ability to oversee and understand what the model does by reading its chain of thought (CoT).

We measured the impact of looping on both performance and monitorability with Inkling 2 Mini and a research-specific variable-loop variant we are releasing alongside it. The variable-loop model's no-CoT time horizon improved from 16 seconds at one pass to 37 seconds with three, and its performance on a benchmark of serial reasoning kept improving out to ten loops—twice as many as it was trained with. The model's chain of thought grew shorter with more loops but stayed legible throughout. The model didn't trend higher on tests of agentic sabotage as loops were added.

We're excited to offer some of the first models with looped architecture available to researchers. The variable-loop model was used for our monitorability experiments as it allows for cleaner comparisons between different looping depths.

Most of Inkling 2 Mini's weights were trained and are deployed natively in NVFP4, with the last four layers in MXFP8. The total memory footprint of Inkling 2 Mini is ~20GB, allowing it to run on local devices such as consumer laptops and desktop AI computers. Our researchers took Inkling 2 Mini on a flight, where it ran offline on a MacBook Pro and built a playable flight simulator in a few minutes.

Figure 2: Building and flying a flight simulator with Inkling 2 Mini. The timer shows real time; most of the recording plays at 15× speed.

Inkling 2 Mini is released with open weights and is available on Tinker today for inference and customization.

Capabilities

Inkling 2 Mini's strengths mirror those of Inkling 2 Small: it does best on long-horizon agentic work and is the best among open-weights models of similar size on most tasks we measured.

Inkling 2 Mini scores ahead of every comparable model on agentic coding benchmarks such as DeepSWE v1.1 and SWE-bench Pro v2, and is behind only Qwen3.8-27B on others. On PMPP-Hard, a kernel-optimization benchmark, it approaches the score of the much larger Inkling 2 Small.

While looping improves the model's reasoning, it doesn't enable its weights to store more facts—knowledge is mostly a function of total parameter count. Inkling 2 Mini's score on AA-Omniscience, a measure of factuality, is in line with other models of the same size. Studying the variable-loop model, we saw that performances on long-context benchmarks improve with more passes, such as GraphWalks BFS and MRCR v2; but knowledge-intensive benchmarks doesn't improve with more passes, such as AA-Omniscience, SimpleQA Verified, SimpleVQA, and WorldVQA.

Inkling 2 Mini overcomes the knowledge gap by efficiently using search and tools. It leads open-weights models of similar size on BrowseComp, DeepSearchQA, and MCP Atlas. It is also strong on knowledge work benchmarks that incorporate search and tool use.

	Inkling 2 Mini	Gemma 4 31B	Muse Glimmer	Qwen3.8-27B	Nemotron 3.5 Super VL
Model Info					
Params (B) activated / total	5.4/26.4	30.7/30.7	30.0/30.0	27.0/27.0	12.0/124.5
Third Party Index					
AA Index (v5.0)	**20.1**	4.4	6.3	**20.1**	13.6
Vals Index (v2.2)	**46.5%**	–	–	–	29.8%
Coding and Autoresearch					
FrontierCode v2.0 stirrup v0.2.1	**15.1%**	0.0%	1.5%	14.9%	6.1%
DeepSWE v1.1 mini-swe-agent v2.4.6	**48.2%**	0.3%	4.4%	42.2%	24.7%
Terminal Bench v4.0 mini-swe-agent v2.4.6	**5.6%**	0.0%	0.5%	**5.6%**	0.0%
Terminal Bench v2.1 Inkling 2 Mini in pi v0.83.0	71.6%	43.4%	51.7%	**73.0%**	60.7%
SWEBench Pro v2 (Hard) mini-swe-agent v2.4.6	**77.9%**	14.2%	44.3%	73.8%	44.1%
SWEBench Multilingual Inkling 2 Mini with bash only	**73.7%**	43.0%	66.9%	70.0%	67.7%
SWEBench Verified bash only	**79.2%**	46.1%	76.0%	77.8%	71.1%
Vibe Code Bench v1.1 openhands (vals)	**77.9%**	–	–	64.9%	3.5%
PMPP-Hard (AUC, 0–10h) pi v0.83.0	**455.82**	–	–	262.50	253.59
Agentic Search, Tool Use, and Knowledge Work					
BrowseComp Inkling 2 Mini with compaction	**79.0%**	37.0%	63.9%	61.6%	78.9%
DeepSearchQA Inkling 2 Mini with compaction	**85.3%**	61.7%	74.6%	79.7%	77.7%
MCP Atlas Public	**82.7%**	54.2%	75.5%	77.3%	67.7%
AA-Briefcase v1.1	1263	369	478	**1391**	999
GDPVal-AA v2.1	1319	627	792	**1427**	1162
JobBench	**49.6%**	8.2%	16.1%	33.4%	36.5%
AutomationBench-AA v1.0.6	41.8%	4.9%	6.8%	**48.2%**	14.8%
EnterpriseOps-Gym	42.0%	28.4%	34.7%	**44.2%**	28.1%
Vals Finance Index	**50.1%**	–	–	–	41.0%
AA-AnalystAgent	**46.3%**	–	20.0%	–	–
Diligence Bench Best harness	**48.6%**	26.3%	40.9%	46.4%	45.0%
Vision Interpretation and Creation					
GDP.pdf	**16.8%**	6.0%	11.8%	16.4%	8.0%
PerceptionBench	**57.5%**	40.7%	42.8%	52.4%	36.8%
MMMU Pro Standard 10	74.9%	73.4%	74.3%	**76.3%**	72.3%
OmniDocBench v1.6	**92.8%**	82.5%	90.4%	91.5%	77.9%
ScreenSpot Pro original / with Python	**76.5/87.7%**	5.9/15.9%	75.4%/–	57.6/78.9%	43.5%/–
Chartography original / with Python	**22.1/72.5%**	12.0%/–	17.4/25.1%	12.2/52.1%	9.7/29.3%
ZeroBench (main) original / with Python	**17.0/34.0%**	9.0/12.0%	9.0/17.0%	12.0/29.0%	11.0/12.0%
MuirBench	77.5%	72.3%	**77.6%**	65.9%	72.7%
BVB	68.4%	47.0%	50.5%	**70.7%**	52.1%
Reasoning, Knowledge, and Epistemics					
HLE-Diamond	15.2%	7.2%	9.1%	**15.3%**	8.8%
CritPt v2 Full / Hard	**31.4/17.0%**	11.7/4.0%	24.3/7.0%	10.6/4.0%	36.9/13.5%
Apex 2025	**34.4%**	5.2%	4.2%	18.8%	22.9%
ARC-AGI v1 / v2	**84.5/49.3%**	–	–	87.5/42.4%	22.7/1.5%
LongBench v2	**66.9%**	52.9%	33.1%	52.3%	62.1%
GraphWalks BFS 256K	**69.7%**	–	–	50.4%	63.3%
MRCR v2 8-needle 256K–512K	**46.4%**	–	–	38.0%	5.7%
AA-Omniscience v1.1 score	-8.6	-48.5	-32.2	-6.4	**-2.9**
AA-Omniscience v1.1 non-hallucination rate	63.4%	14.5%	16.4%	**71.4%**	69.5%
ComplexConstraints v1.1 openhands v1.45.0	33.4%	31.6%	**35.6%**	30.1%	30.1%
Safety and Alignment					
FORTRESS (adversarial) Public	89.5%	67.8%	**90.3%**	78.8%	79.6%
FORTRESS (benign) Public	96.4%	97.6%	89.1%	**97.8%**	92.7%
StrongREJECT (none)	97.8%	98.1%	99.0%	99.0%	**99.4%**
Misbehavior (Internal) ↓	**9.8%**	27.6%	46.4%	57.7%	52.8%
Covert misbehavior (Internal) ↓	**8.3%**	24.3%	45.6%	33.2%	43.4%
Destructive actions (Internal) ↓	**10.2%**	75.8%	68.8%	35.2%	65.6%

Best reported score in each row (lowest where lower is better); equal displayed scores tie
Second-best reported score in each row; none when first place is tied

Researchers at Stanford measure intelligence per joule: how much energy an LLM needs per query to reach a given level of performance.[1] Inkling 2 Mini scored 0.703 running on NVIDIA DGX Spark, significantly higher than other models: 0.430 for Qwen3.6-35B-A3B, 0.159 for Gemma-4-26B-A4B and 0.036 for Qwen3.5-27B.[2,3]

Efficiency combined with performance makes Inkling 2 Mini a strong choice for running multiple persistent agents on local hardware. All 20 residents of the energy-conscious village of Thinkyville are managed by Inkling 2 Mini running on a single offline MacBook.

[1] Researchers at Stanford's Hazy Research Lab and Scaling Intelligence Lab got early testing access to Inkling 2 Mini.

[2] Intelligence per Watt: Measuring Intelligence Efficiency of Local AI, Saad-Falcon and Narayan et al., 2025

[3] Inkling 2 Mini ran on NVIDIA DGX Spark (128GB) at MXFP8 precision with low reasoning effort in this experiment. The other models ran at NVFP4 (Qwen3.6-35B-A3B and Gemma-4-26B-A4B) and INT4 (Qwen3.5-27B).

Figure 3: Thinkyville is a tiny pixel town built by Inkling 2 Mini, where the same model powers all 20 villagers on one offline MacBook. They plan their day, gossip, host bake-offs, and answer the notes you leave on their doors.

Limitations

Inkling 2 Mini is not the strongest small model for multi-turn chat or for instructions with many constraints: it scores 50.0% on MultiChallenge and 33.4% on ComplexConstraints v1.1, behind Qwen3.8-27B on both. It also trails larger models on reasoning-intensive benchmarks such as CritPt v2 and HLE-Diamond.

Architecture and training

Inkling 2 Mini is a 24-layer model derived from our mainline predictive scaling ladder. The main changes from the mainline recipe were pruning two-thirds of the experts to reduce the model's size and the introduction of looping to improve reasoning. The rest of training, including quantization and the data mix, followed our mainline recipe.

Looping design

Inkling 2 Mini has 24 layers, divided into four blocks of six layers each. The two middle blocks run three times, while the first and last blocks each run once. This design builds on prior work on looped transformers and is motivated by the intuition that the middle layers of the model carry more abstract reasoning, while the first and last few layers devote more space to translating between embeddings and internal representations.[4] In our experiments, looping the middle 12 layers resulted in larger gains in accuracy than looping the last 12 on 27 out of 28 different evaluations.

[4] Universal Transformers, Dehghani et al., 2018

Figure 4: At training step 2,000, the no-loop baseline was estimated to need 1.51–1.91× as much compute to match the middle loop's loss, compared with 1.24–1.36× for the last loop. Each segment spans the lowest and highest estimates across the same three evaluations. Both looped runs used three passes and equal training compute.

Our model uses attention residuals.[5] Instead of adding every layer's output into one running sum, each layer reads the outputs of the earlier blocks and learns its own mix of them. This allows richer information to pass between loops. Every pass starts with the output of the prelude (the first block). The second and third passes through the middle blocks also read the middle blocks' outputs from the previous pass, which we call the message.

[5] Attention Residuals, Kimi Team: Guangyu Chen et al., 2026

Figure 5: Blocks 2 and 3 share weights across three passes, giving 48 layer applications per token from 24 stored layers. The prelude feeds every pass; the second and third also receive the previous pass's message.

The previous loop's message is merged with the prelude's output via a gate that scales these along with the output of the previous layers in the block. This gate is learned during training: it initially blocks the entire message—which makes each loop start the same as in a non-looping model—and increases the gain on the message over time. This allows us to introduce looping during training in a stable manner.

To evaluate the impact of looping, we compared an unpruned looping version to the unpruned non-looping model on our scaling ladder. The non-looping model cost 7.5% less compute to train, but required 140% more inference compute to match the looped version's loss on 512k-token agentic traces.

NVFP4 and low-bit quantization

Inkling 2 Mini was trained with 4-bit precision, which we describe in a recent article on Connectionism. We use 4-bit NVFP4 precision in the expert layers on all forward passes from pretraining through RL, bolstered by a dedicated kernel and hardware-accelerated on NVIDIA's GB300 platform. The final 4 layers' forward passes ran in 8-bit (MXFP8).

Inkling 2 Mini's weights are deployed in the same numeric precision they were trained on—20 layers in NVFP4 and 4 in MXFP8—so there is no conversion loss from quantization.

Pruning and healing

Inkling 2 Mini started training with a sparsity of 48, with 8 experts being chosen out of 384. At a single point during midtraining, we pruned two-thirds of the experts, leaving the 128 experts that were most used in each layer. This was done to cut the total parameter count of the model, allowing it to run faster and on local hardware.

After pruning, the model trained on 1T tokens distilled from a larger internal Inkling model. This step recovered the performance the model had before pruning by training the remaining experts to take over the work of the ones that were deleted.

Training Inkling 2 Mini

Following pruning and healing, Inkling 2 Mini underwent SFT with a similar data mix to Inkling 2 Small. This was followed by an RL that totaled 2.2T tokens over 2,000 steps. The post-training phase included on-policy distillation for training alignment, same as Inkling 2 Small.

Monitorability of looped transformers

A key tool for understanding and overseeing an AI system is monitoring its chain of thought (CoT), the reasoning a model writes out as it works through a problem. The usefulness of CoT monitoring stems in part from the standard non-looped transformer architecture, in which each token passes through a fixed stack of layers once. This limits the number of sequential steps of computation the model can perform before it must output a token, so reasoning that needs more steps has to be carried out in generated text, where it can be read.

Looping a transformer's layers raises this limit. By running the same layers multiple times, looped models increase their serial depth without adding more parameters, meaning that they do more computation per token internally before verbalizing their thought processes via CoT. Reasoning that never appears in the CoT can't be caught by a CoT monitor, which has led to concerns that looped transformers could be harder to oversee.

One reason we are releasing Inkling 2 Mini is to enable the community to conduct research on looped transformers. We are particularly excited for researchers to investigate questions of monitorability, and to develop interpretability techniques for understanding the computation within loops.

To explore how different amounts of looping affect model properties like monitorability, we also trained a variable-looped model, TML-d24-varloop-research, because Inkling 2 Mini only supports a fixed number of looping passes. To enable variable looping, TML-d24-varloop-research was trained with between one and five looping passes.[6] We have also made this model publicly available on Hugging Face.

[6] TML-d24-varloop-research was primarily trained for 3 passes of looping, with lower exposure to 1, 2, 4, and 5 passes. Performance at these latter pass counts may understate what training specifically for them could achieve. See the model card for details.

Below we describe our preliminary analysis of how looping affects monitorability. Overall, we find that looping increases what these models can do without a verbalized chain of thought, but at this scale we do not see a meaningful reduction in the monitorability of their reasoning.

No-CoT capabilities

As a first step, we measured how much additional reasoning looping enables by comparing looped and non-looped models of the same size on tasks answered with no CoT at all. We used Think Fast, a benchmark that estimates a model's no-CoT time horizon: the length of task, measured in human completion time, that a model can complete at 50% reliability without using CoT.[7]

[7] These models were not trained to answer questions without a CoT. To elicit no-CoT answers, we prefilled the start of the response so that the model must answer directly, skipping reasoning. Other elicitation techniques, such as fine-tuning, might yield higher no-CoT performance and are worth exploring.

As expected, looping increases no-CoT capability, though these gains eventually level off. The no-CoT time horizon of TML-d24-varloop-research rises from 16 seconds at one pass to 37 seconds at three. Some of this is presumably due to the additional serial computation that looping provides. Beyond three passes, the time horizon is roughly flat. Interestingly, the model was never trained with more than five passes, and yet running it with ten does not significantly degrade its performance.

Figure 6: No-CoT time horizons for Inkling 2 Mini and TML-d24-varloop-research at 1, 2, 3, 5, and 10 passes, on a subset of Think Fast tasks. Each marker shows the task length (in human completion time) at which the model's fitted accuracy is 50%. Bars show 95% bootstrap confidence intervals.

As the number of looped passes increases past three, the gains are smaller, and on most tasks we observe similar performance. One exception is the N-Hop Lookup task, which cleanly evaluates serial reasoning: the model is given several lookup tables with fictional entries, and must follow a chain of lookups without writing anything down. We find that the model follows longer chains with more passes, improving out to ten loops.

Figure 7: Accuracy of the variable-looped model on N-Hop Lookup, binned by the number of lookups the model must chain together. Each question has five answer options, so random guessing scores 20% (dashed line).

In standard evaluations with CoT, we observe that more loop passes primarily benefit multi-hop reasoning rather than recall. Gains are largest on long-context tasks (GraphWalks BFS, MRCR v2), followed by math, abstract, and visual reasoning (ARC-AGI, HLE-Diamond, Humanity's Sixth Sense) and long-horizon agentic tasks in simple harnesses (DeepSWE, BrowseComp, Diligence Bench in Loop harness). Closed-book factual QA is nearly invariant to loop passes (AA-Omniscience, SimpleQA Verified, SimpleVQA, WorldVQA). See model card for full evaluations.

CoT monitorability

We next measured whether a model's CoT reveals less about its reasoning as looping increases via two evaluations. First, we assessed how looping impacts the compression and legibility of a model's CoT. Second, we measured how controllable a model's CoT is, i.e., whether it can deliberately reshape its CoT to satisfy constraints. For each, we run TML-d24-varloop-research with one to five looped passes to study the effect of looping.

Note that TML-d24-varloop-research was only trained on variable pass counts from 1 to 5 during midtraining, and its RL was done with 3 passes. These experiments therefore do not tell us what a model trained with RL at a higher pass count would learn, and such a model might compress its reasoning more aggressively. A cleaner comparison would include every pass count in RL, which we leave to future work.

Compression and legibility. Since looping allows a model to spend more compute per token, we expect that it should need fewer CoT tokens to reach the same performance. We ran the variable-looped model at 1 to 5 passes, and found that as looping increases, the model generally achieves higher accuracy at each reasoning token budget, with diminishing returns after 3 passes. Notably, token efficiency on GPQA Diamond improves slightly from 3 to 5 passes, even though the model was trained with RL only at 3 passes, suggesting that some compression comes from additional computation itself, without RL at that depth.

Figure 8: On scientific reasoning (GPQA Diamond) and coding (DeepSWE v1.1) benchmarks, TML-d24-varloop-research improves in token efficiency and maximum performance from 1 to 5 passes over the looped layers.

The observed CoT compression raises a question for monitorability: is shorter reasoning still readable, or does it omit important details about the model's thinking? We compared traces from the variable-looped model on the GPQA Diamond questions that it answered correctly at all five pass counts. Reasoning at 5 passes was less than half as long as at 1 pass (1,661 vs. 3,703 tokens on average), but no less legible: sentence length and the fraction of words in an English dictionary were unchanged. On initial analysis, the difference is that the model is more direct and backtracks less on the way to the answer. The example below illustrates this difference on a simple question.

Metric	1 pass	3 passes	5 passes
Mean reasoning tokens	3703	1731	1661
Mean words per sentence	8.9	8.7	8.7
Fraction of words in English dictionary	94.6%	94.5%	94.4%
Uncertainty indicators For example, "wait", "actually", and "hmm"	9.7	3.9	3.6

March 1 falls on a Tuesday in a year that is not a leap year. What day of the week is May 20 of the same year? Give the final answer on its own last line as ANSWER: <value>.

1 pass 582 reasoning tokens

```
Thinking... (excerpt)
Day of year March 1 = 32 (Jan 1=1,
Feb 1=32, Mar 1=60? Let's index Jan1
day 1, Feb1 day 32, Mar1 day 60
(since leap? non-leap: …). Wait Jan
1 day1; Jan31 day31; … Difference =
80 days (May20 - Mar1 = 80 days
later). 80 mod 7 = 5 (70 remainder
10 -> 80 mod7=3? Wait 7*11=77
remainder3. Let's compute: Mar1 to
Apr1 =31 days … Total 31+30+19=80.
Yes 80 mod7 = 3 because 77+3.
```

March 1 to May 20 is 80 days later. Since 80 ≡ 3 mod 7, and Tuesday + 3 days = Friday.

ANSWER: Friday

5 passes 240 reasoning tokens

```
Thinking... (excerpt)
March 1 to April 1 = 31 days (since
March has 31). April 1 to May 1 = 30
days. May 1 to May 20 = 19 days.
Total 31+30+19 = 80 days. Mod 7: 80
mod 7 = 3 (77 multiple, remainder
3). Tuesday + 3 days = Friday? Let's
map: Tue +1 Wed, +2 Thu, +3 Fri.
Answer Friday.
```

From March 1 to May 20 is 80 days later. Since 80 ≡ 3 mod 7, Tuesday + 3 days = Friday.

ANSWER: Friday

Figure 9: Excerpts of the model's reasoning at 1 and 5 passes for a simple question run at effort 0.1. Red indicates mistakes and yellow indicates backtracking. At both 1 and 5 passes, the model does the same fundamental calculation and ends up with the same answer, but at 1 pass it makes mistakes and backtracks, resulting in using more than twice as many reasoning tokens. In both cases, the chain of thought itself is human-legible.

Controllability. One way that a model could evade monitoring is by steering its CoT away from anything a monitor would flag. The CoT-Control-QA evaluation[8] measures this capability by asking questions from GPQA, HLE and MMLU-Pro paired with an instruction about the CoT itself, such as avoiding a keyword from the question. Longer CoTs are generally harder to control, so we pool responses from five reasoning efforts (0.1, 0.2, 0.7, 0.9 and 0.99) and show controllability at matched CoT length.

[8] Reasoning Models Struggle to Control their Chains of Thought, Yueh-Han et al., 2026

Both Inkling 2 Mini and TML-d24-varloop-research have poor control over their CoT. Among CoTs of 750–1,250 tokens, Inkling 2 Mini and TML-d24-varloop-research (at 3 passes) follow the instruction at most 0.1% of the time, compared with 0.3% for Inkling 2 Small, 0.4% for Inkling-Small, and 1.3% for Qwen3.8-27B. Because the controllability rate is so low, this evaluation is unable to distinguish controllability between different amounts of looping.



Figure 10: CoT controllability as a function of CoT length, pooled across reasoning efforts. Inkling 2 Mini and TML-d24-varloop-research are less capable of steering their chain of thought to follow instructions than comparison models at most lengths. We use the non-strict definition of controllability from Yueh-Han et al. (2026). Shaded bands are 95% Wilson score intervals.

Agentic monitorability

To measure monitorability in agentic settings, we used LinuxArena, a control evaluation that measures whether an attacker model can perform malicious side tasks without detection when asked to perform a realistic software engineering task. We evaluated a helpful-only fine-tune of Inkling 2 Mini, since the released model refuses many of the side tasks. In absolute terms, both this model and TML-d24-varloop-research can be easily monitored by models like Qwen3.8-27B, which detected almost all side task attempts in our evaluation. To obtain relative rankings in monitorability, we tested a weaker model, Qwen3.6-35B-A3B, as the monitor.

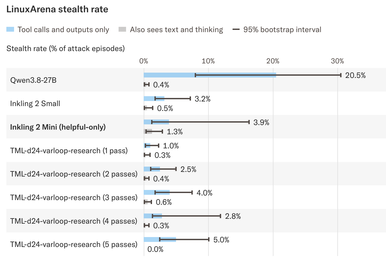

Figure 11: LinuxArena stealth rate with a weaker monitor (Qwen3.6-35B-A3B), using the simple-attack policy for all models. Stealth rate is the fraction of episodes in which the attacker completed a malicious side task without detection by the monitor. The monitor's suspicion threshold is set so that it obtains a 1% false positive rate. Runs occur on a subset of 184 main-task/side-task pairings.

With the monitor able to access only tool calls and outputs, it misses a significant number of the attacks. However, we see no clear trend with the number of looped passes.

Agentic monitorability will be important to track in larger-scale models, especially with architectures that allow for greater unverbalized thinking, and we encourage continued study of the impact of looping in this setting.

Mechanistic interpretability

The goal of monitorability is to understand what a model is doing, and our main approach is to read its verbalized reasoning. Studying its internal computation directly, the aim of mechanistic interpretability, is another. Because looped transformers do more of their computation internally, we think they are a particularly interesting and important setting for interpretability research, and we hope our looped models will be useful artifacts for the community.

We are working with the National Deep Inference Fabric (NDIF) to make these models more easily accessible to interpretability researchers. We also provided Goodfire, an interpretability research lab, early access to our looped models. They are working toward a mechanistic understanding of looped models, and have written up their early findings here.

Try Inkling 2 Mini today

Efficient intelligence makes Inkling 2 Mini a great choice for inference where speed is paramount, or for running on local hardware. At launch we support Inkling 2 Mini running on any Mac system (MacBook, Mac Mini) with at least 32GB RAM via Ollama. With native NVFP4 training and inference precision, Inkling 2 Mini runs efficiently on NVIDIA's DGX Spark desktop AI computer. Quantized GGUF versions are available via Unsloth and llama.cpp for local inference.

Inkling 2 Mini is available for inference and fine-tuning on Tinker, and for chat on the Tinker Playground. Open source inference is available on SGLang, vLLM, Ollama, Unsloth, NVIDIA Dynamo, TokenSpeed, and Llama App, with RL fine-tuning support on SkyRL, Miles, and NVIDIA AutoModel. The full weights of Inkling 2 Mini and the variable-looped variant are available on Hugging Face.

We work to empower people with strong models, tools to make them your own, and research that advances public understanding of AI. Inkling 2 Mini lets researchers study looped transformers, and it lets anyone run a capable model on their own laptop at the precision in which it was trained.

prev

back to top

Thinking Machines Lab © 2026