94% of Frontier Retrieval Quality from a 494M-Parameter Model

Method summarisation for code knowledge graphs in 2.43 GPU-hours,
and the point where more training data stops paying

Akhil Katakam
KoraGraph
github.com/agilkatakam

August 2026

Abstract

Code knowledge graphs need one short sentence per method saying what that method is for. The sentence is what developer queries are matched against, so its quality is retrieval quality, and at the fourteen thousand or so method vectors a real repository carries, it is produced at the scale of the codebase rather than of usage. Sending every method body to a hosted endpoint to obtain it is not available to every buyer at any price.

We fine-tuned Qwen2.5-Coder-0.5B-Instruct, 494 million parameters, on 20,000 teacher-labelled examples for 2.43 hours on a single T4, roughly $0.85 of commodity GPU time, and measured it against four hosted models and its own untrained baseline on a sealed corpus of 2,393 methods with repository-cluster-disjoint splits, 2,320 of which every system answered and which all reported figures use. Scored at top-8, which is what the production retriever uses, the fine-tuned model reaches 87.5 against the teacher's 93.3, or 93.8% of frontier quality. It beats Gemini 2.5 Flash Lite, a hosted model KoraGraph runs in its own extraction path, by 4.1 points with $p < 0.0001$ on a paired test. DeepSeek V4 Flash leads the table at 93.9, ahead of the teacher. The local model sits 6.3 and 5.8 points behind that pair, significant on paired tests.

Two further results. First, 91% of the total gain over the untrained baseline arrives within the first 2,500 training examples; going from 2,500 to 20,000 buys 1.7 points, and no individual step in that range is significant, while held-out loss keeps improving by 14% across the same span. The training objective goes on rewarding data long after the task stops responding to it. Second, the evaluation protocol determines the answer. Scoring a single model-written sentence against a pool of teacher-written ones, which is the natural first implementation, reads 2.7 points below the uniform protocol a deployed graph actually requires. We give the protocol and the measurement that separates the two.

The corpus and the generated sentence sets are not being published at this time.

1 Why this measurement exists

KoraGraph builds a persistent, queryable representation of a repository and answers developer questions against it. Every method in that representation carries a *Purpose*, a single sentence stating what the method does for the system. The sixty to a hundred and sixty character band used throughout this paper is the corpus specification, tighter than what the deployed prompt asks for. That sentence is concatenated with the method's node type, class and name, together with whatever structured fields the extractor recovered, to form the passage that gets embedded, and a developer's question is matched against those passages. The Purpose is the only free-text field in that passage and the only one that carries intent, so a vague sentence costs recall that nothing

downstream restores. We hold every other field fixed and vary only the Purpose, which isolates its contribution and does not reproduce the full production passage.

Two facts about this call make it worth optimising separately from everything else the product does. It produces one artefact per method rather than per question, and a measured repository in our own corpus carries 14,714 active method vectors, so the volume is set by the size of the codebase and not by usage. Production currently obtains these sentences a file at a time inside a larger extraction call; a dedicated model would be invoked per method. And it is close to a pure text-to-text task with a short output, which is exactly the shape that sequence-level distillation into a small model has historically handled well [14, 11].

The reason to want it local is not cost. A full pass over thirteen thousand methods with DeepSeek V4 Flash costs about forty-six cents at the rates we metered, which is not a number that decides anything. The reason is that a large class of buyers cannot send source code to a third-party inference endpoint at all. Export control, data residency obligations, customer contracts and internal policy each independently rule it out, and for an organisation under any of them the hosted column of our results table does not exist as an option. That this class of buyer is large is our commercial read and not a finding of this paper; the technical setting is treated in [1]. A second reason appears once a graph is treated as a durable asset and not a cache: an index built by a hosted endpoint is not reproducible, because the endpoint is versioned by its vendor and will not answer identically in a year. A pinned checkpoint will.

So the question this paper answers is narrow. If you decline to send your code anywhere, how much retrieval quality do you give up?

2 The task

Given a method and its surrounding context, produce one sentence. The specification below is what the four hosted comparison systems were given. It is a condensed form of the specification used to elicit the training labels from the teacher. The fine-tuned arms receive no specification, having learned the format from examples, and the zero-shot baseline receives the corpus instruction string in its place.

> One sentence, sixty to a hundred and sixty characters, ending in a full stop. Say what the method does for the system, the job it performs and on what, rather than restating its signature. Every identifier named must appear verbatim in that method's own source. Never invent a mechanism, collaborator, queue, cache, table, protocol, retry policy, metric or side effect the source does not show. No line numbers, no file paths, no "this method" preamble. Third person, present tense.

This is a document expansion problem in the sense of Nogueira et al., where generated text is attached to an item to make it retrievable by queries whose vocabulary the item does not contain [15]. It differs from the classical setting in that the expansion is a human-readable artefact shown in the product, so it is constrained by a format and a grounding rule as well as by retrieval performance, and in that the underlying items are code rather than prose [13].

The metric follows from deployment. The production retriever fuses a dense and a sparse channel from BGE-M3 [5, 3] in roughly a seven-to-three ratio, alongside a third lexical leg that this harness does not model, and returns eight passages. We therefore report hit@8 as the primary figure: the fraction of methods whose own passage appears in the top eight results for that method's developer question, against a pool of competing methods. hit@1 is reported alongside as a strictness check, not as the headline, because a system that lands the right method at rank three has done its job in production.

3 The system under test

3.1 Model and training

The student is Qwen2.5-Coder-0.5B-Instruct [12, 19], 494,032,768 parameters, 24 layers, hidden size 896, tied embeddings, Apache 2.0. We fine-tune all parameters. No adapters, no quantisation during training. Sequence length 640 tokens, learning rate 1×10^{-5}, effective batch size 16 by way of two per device and eight accumulation steps, fp16, seed 20260813. Loss is masked to the completion span only, with the end-of-turn token inside the loss and the padding token distinct from it.

Serialisation is the failure mode that kills fine-tunes silently, so training and every evaluation import the input builder from one module and neither reimplements it. The chat template contributes 24 tokens before the input, 5 between input and target, and 2 after; treating those 31 as a flat prefix masks the first two completion tokens of every example, which we verified against 4,000 records before fixing.

Held-out loss during training is computed on a fixed 200-record subset of the validation split, so the loss figures reported below rest on $n = 200$ and support only the within-arm comparison they are used for. Training ran on a single T4. The 20,000-example arm took 2.43 hours and 10.2 GB of device memory, and all five arms together 4.74 T4-hours. Priced at a public on-demand T4 rate of $0.35 per GPU-hour, that is $0.85 for the shipping arm and $1.66 for the entire study; at the $0.20 median across commodity providers, $0.49 and $0.95. These are modelled from the measured wall-clock, not metered charges.

3.2 Corpus and splits

The teacher is a frontier code model, which produced 24,650 labelled records over permissively licensed open-source repositories. Splits are disjoint at the level of repository clusters, not files, so no repository contributes to both training and test, and near-duplicate bodies are collapsed before splitting. Training arms are nested prefixes of a single deterministic stream built by a largest-deficit rule applied at both language and repository level, so the 2,500-example arm is a strict subset of the 5,000-example arm and so on up. That nesting is what makes the data-scaling comparison in Section 6 a controlled one rather than five independent samples.

The test split was held out from the first arm onward and was not scored until every training run in this paper had finished.

3.3 Comparison systems

Four hosted models were given that specification and each method's source, name, class and language, batched eight or ten per request: DeepSeek V4 Flash [7] and Nemotron 3 Super 120B-A12B [16] at eight, Gemini 2.5 Flash Lite [9] and Claude Haiku 4.5 [2] at ten. Gemini 2.5 Flash Lite is a model KoraGraph itself runs in its extraction path, which makes it a commercially meaningful comparison. The teacher is included as a fifth system, scored on the same sealed test set it never saw during label production for that split.

Two transport-level accommodations were necessary and are reported because they affect what the numbers mean. DeepSeek V4 Flash and Nemotron are reasoning models. At their default settings Nemotron emitted 18,493 characters of reasoning per batch and reached its length cap before writing a single sentence, yielding 8 usable outputs across 21 billed calls, and DeepSeek at its default spent roughly 4,400 reasoning tokens per batch at $0.0026 per call, pricing one pass over the test set at $0.76 against $0.08 with reasoning disabled, at roughly a twelfth of the throughput.

Language	Train	Val	Test	Language	Train	Val	Test
javascript	1.4	6.3	15.1	swift	3.5	16.1	4.2
rust	11.1	0.0	11.9	elixir	1.3	15.2	3.8
scala	1.3	0.0	11.7	typescript	9.6	1.4	3.1
java	13.7	9.9	10.4	ocaml	1.8	0.0	2.5
go	3.9	8.8	9.0	csharp	10.2	0.0	2.0
python	6.4	9.0	7.1	cpp	0.9	0.0	2.0
php	13.7	17.7	6.3	objc	0.5	0.0	1.2
c	1.7	11.6	4.6	kotlin	11.4	1.4	0.3
ruby	5.8	0.0	4.3	others (5)	1.9	2.7	0.7
Records: 20,000 train, 1,108 val, 2,393 test				Repositories: 49, 9, 45			

Table 1: Corpus composition by language, percentage of each split, across 21 languages. The three splits are disjoint at repository-cluster level, which is why their language mixes differ: whole repositories move together. The validation split in particular is not a scaled copy of training, and that is the composition bias noted in Section 5.

System	Configuration	Calls	Prompt tok	Completion tok	Spend
Gemini 2.5 Flash Lite	default	240	489,947	209,375	$0.133
Nemotron 3 Super	reasoning off	269	415,063	78,814	$0.112
DeepSeek V4 Flash	reasoning off	300	452,457	120,609	$0.083
DeepSeek V4 Flash	default, partial	19	26,498	173,335	$0.048
Ours, five arms	local training			4.74 T4-hours	$1.66
Frontier teacher	labels reused			n/a	$0

Table 2: Spend for the comparison runs. Hosted figures are metered, read from the gateway's own usage reporting rather than modelled from list rates; the training figure is modelled from measured wall-clock at $0.35 per T4-hour. The nineteen default-configuration DeepSeek calls returned 80 usable sentences before the run was stopped, and their completion-token count against their prompt-token count is the reasoning overhead that motivated disabling it. Claude Haiku 4.5 was generated before per-run ledgering was added and is not included, so this is not a complete total for the paper.

Both were therefore run with extended reasoning disabled, at which point Nemotron returned four of four parsed outputs in three seconds per batch. DeepSeek's provider-side JSON mode also emitted a malformed prefix on nearly every reply and was disabled for that model. The prompt was not altered for any system.

The untrained baseline is the same Qwen2.5-Coder-0.5B-Instruct checkpoint, zero-shot, given the specification in its prompt and truncated to one sentence. Without the specification the base model writes multi-paragraph prose and the comparison measures format guessing instead of the task; without the one-sentence cap it receives several times the passage budget every other system is held to.

4 How retrieval is scored

4.1 Uniform treatment

The pool a query competes against must be written by the system under test. This is not a stylistic preference. In a deployed graph, every passage is produced by the same model, so if that model's sentences are systematically less distinctive from one another they blur together and hit@k falls, and if they are more distinctive it rises. Scoring a single model-written sentence against a pool of teacher-written ones measures neither effect.

Both protocols were run, on the validation split at $n = 300$ queries against a pool of 1,108. Under mixed treatment, one model-written purpose competing against 1,107 teacher-written ones, our student reads 90.0 against the teacher's 94.7. Under uniform treatment, with the pool rewritten by the system under test, the pair reads 92.7 against 92.7. The two runs draw their query samples independently, so this is not a paired comparison, and the teacher's own 2.0-point movement between them bounds the sampling term: the teacher's pool is teacher-written under either protocol, so for the teacher alone the two runs are the same experiment. Netting that out leaves roughly 2.7 points attributable to treatment. The mixed figure is an artefact of the comparison design rather than a property of either system, and the size of it, large enough to reverse a conclusion, is why the protocol is stated here rather than assumed. Every number in this paper is uniform-treatment.

A second correction follows from the same principle. Systems differ in coverage: DeepSeek returned 2,345 of 2,393 methods and Gemini 2,370, while the local models returned all of them. An empty purpose is not a neutral pool member, it is a uselessly weak distractor, so a system that skipped methods would compete against its own blanks and score higher for having answered less. We therefore restrict the pool itself, not merely the query set, to the 2,320 methods every system answered. All eleven systems are scored on identical passages and identical queries.

4.2 Ranking and statistics

Rank is the count of pool members scoring strictly above the target, so rank zero is a top-1 hit and ties do not promote the target. Confidence intervals on hit@8 are Wilson intervals. Comparisons between systems use an exact McNemar test on discordant pairs, because all systems are scored on the same queries and two overlapping Wilson intervals do not license a claim of no difference.

5 Results

5.1 The comparison

Eleven systems, 2,320 methods, uniform treatment, hybrid dense-plus-sparse retrieval.

Table 4 gives the same eleven systems across every k we computed, and Table 5 the output-quality columns. Format is the percentage of outputs satisfying the length, single-sentence and terminal-punctuation rules. Ungrounded is the percentage naming an identifier absent from the method's own source, which is the mechanically checkable half of fabrication.

Three readings. The fine-tuned 0.5B model reaches 93.8% of the teacher's hit@8 and 93.3% of DeepSeek's. It beats Gemini 2.5 Flash Lite, a hosted model KoraGraph runs in its own extraction path, by 4.1 points. And fine-tuning is worth 18.7 points over the same weights untrained, which is larger than the entire spread across the five hosted systems, 10.4 points.

System	Class	hit@1	**hit@8**	95% CI	Format	Ungr.
DeepSeek V4 Flash	hosted	67.2	**93.9**	[92.8, 94.8]	93.6	1.1
Frontier teacher	hosted	69.4	**93.3**	[92.2, 94.3]	98.6	0.3
Nemotron 3 Super	hosted	61.6	**90.2**	[88.9, 91.3]	97.8	0.8
Claude Haiku 4.5	hosted	58.1	**89.3**	[87.9, 90.5]	89.9	0.8
Ours, 20k	**local 0.5B**	55.9	**87.5**	[86.1, 88.8]	97.7	0.7
Ours, 10k	local 0.5B	55.8	86.2	[84.7, 87.6]	97.7	1.1
Ours, 5k	local 0.5B	56.3	86.0	[84.6, 87.4]	97.1	0.7
Ours, 2.5k	local 0.5B	55.3	85.8	[84.3, 87.2]	97.6	1.3
Ours, 2.5k one epoch	local 0.5B	54.0	84.9	[83.4, 86.3]	97.1	0.9
Gemini 2.5 Flash Lite	hosted	51.2	**83.5**	[81.9, 84.9]	95.4	0.7
Base 0.5B, zero-shot	local 0.5B	37.2	**68.9**	[67.0, 70.7]	87.7	1.3

Table 3: Eleven systems on the 2,320 methods every system answered, uniform treatment, hybrid dense-plus-sparse retrieval. Intervals are Wilson at 95%.

System	Class	hit@1	hit@3	hit@5	**hit@8**	hit@10	MRR
DeepSeek V4 Flash	hosted	67.2	85.3	90.6	**93.9**	94.7	0.775
Frontier teacher	hosted	69.4	85.9	90.4	**93.3**	94.2	0.787
Nemotron 3 Super	hosted	61.6	80.4	86.2	**90.2**	92.0	0.726
Claude Haiku 4.5	hosted	58.1	78.8	84.5	**89.3**	90.9	0.699
Ours, 20k	**local 0.5B**	55.9	75.9	83.0	**87.5**	89.4	0.677
Ours, 10k	local 0.5B	55.8	76.2	82.0	86.2	88.2	0.675
Ours, 5k	local 0.5B	56.3	75.6	81.0	86.0	87.9	0.676
Ours, 2.5k	local 0.5B	55.3	74.3	80.7	85.8	87.6	0.668
Ours, 2.5k one epoch	local 0.5B	54.0	72.8	79.4	84.9	86.6	0.654
Gemini 2.5 Flash Lite	hosted	51.2	71.0	78.3	**83.5**	85.5	0.633
Base 0.5B, zero-shot	local 0.5B	37.2	54.9	62.1	**68.9**	72.2	0.487

Table 4: Rank profile across every k computed, with mean reciprocal rank. Production uses $k = 8$.

5.2 Paired comparisons

DeepSeek and the teacher are not distinguishable on this corpus, and neither are Nemotron and Haiku. Everything across those two tiers is. Our model sits below both hosted tiers and above Gemini, and each of those placements is significant on a paired test.

6 How much training data is actually needed

Because the arms are nested prefixes, the effect of adding data is measured on one stream and not across five samples. Figure 2 shows the curve and Figure 3 the marginal return behind it.

The first 2,500 examples carry 91% of the total gain, comparing arms at matched epoch count. The remaining 17,500 carry 1.7 points between them. No individual step in that range reaches significance, the closest being the final doubling at $p = 0.053$, but the cumulative 2,500 to 20,000 difference does, at $p = 0.019$ on 159 wins against 119. Eight times the data buys a real 1.7 points. The claim here is about effect size, not about the absence of an effect. Format compliance saturates even faster, moving from 87.7% to 97.6% at 2,500 examples and then staying flat. Ungrounded-identifier rates sit between 0.7% and 1.3% across all five fine-tuned arms with no trend, and we did not test them.

System	Coverage	Format valid	Ungrounded	Mean chars
Frontier teacher	100.0	98.6	0.3	89.0
Nemotron 3 Super	100.0	97.8	0.8	111.7
Ours, 20k	100.0	97.7	0.7	89.4
Ours, 10k	100.0	97.7	1.1	88.9
Ours, 2.5k	100.0	97.6	1.3	89.1
Ours, 5k	100.0	97.1	0.7	88.8
Ours, 2.5k one epoch	100.0	97.1	0.9	86.9
Gemini 2.5 Flash Lite	99.0	95.4	0.7	88.0
DeepSeek V4 Flash	98.0	93.6	1.1	117.9
Claude Haiku 4.5	99.9	89.9	0.8	75.9
Base 0.5B, zero-shot	100.0	87.7	1.3	120.7

Table 5: Output quality, all figures as percentages except the last column. Coverage is of the full 2,393-method test set; every other column is computed on the 2,320 scored in common. Ordered by format compliance.

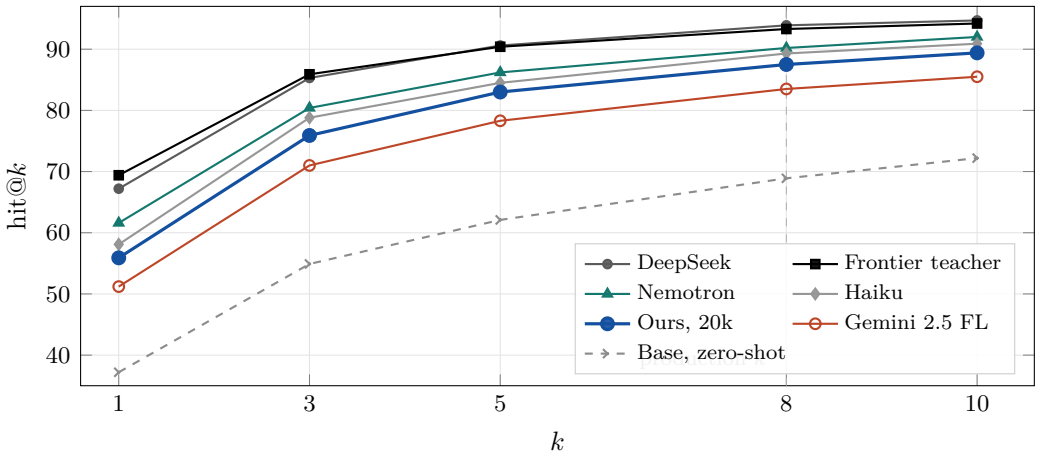

Figure 1: Rank profile across k. The fine-tuned model tracks the hosted systems in shape and sits below them in level, and the untrained checkpoint is separated from all of them at every k. The dashed line marks the k the production retriever actually uses.

Held-out loss tells a different story over the same range, plotted in Figure 4.

Epoch-1 held-out loss falls monotonically with data, by 14% from 2,500 to 20,000 examples, over the same span in which retrieval moves 1.7 points. A practitioner watching validation loss would read that as data paying at full rate, while the deployed metric moves an order of magnitude less.

The loss column carries two caveats. The validation split was built for within-run checkpoint selection and its language composition differs from training by an L_1 distance of 1.109, which is not common-mode across arms, and its 2,500 endpoint is a one-epoch run on a one-epoch schedule. The direction holds; the 14% is indicative.

The per-epoch column carries a second observation. Best held-out loss is at epoch one for every arm with a recorded curve, and every checkpoint scored in this paper is the final-epoch checkpoint, so all three-epoch arms are being scored past their own loss minimum. That is uniform across arms and does not advantage any of them, but it means the epoch-1 checkpoints are an untested and free source of possible improvement.

Comparison	Δ hit@8	Wins	Losses	p
DeepSeek vs. teacher	+0.6	101	88	0.383
Nemotron vs. Haiku	+0.9	132	111	0.199
Teacher vs. ours (20k)	+5.8	209	75	< 0.0001
DeepSeek vs. ours (20k)	+6.3	215	68	< 0.0001
Nemotron vs. ours (20k)	+2.6	191	130	0.0008
Haiku vs. ours (20k)	+1.7	181	141	0.030
Ours (20k) vs. Gemini	**+4.1**	**235**	**141**	$\mathbf{< 0.0001}$

Table 6: Exact McNemar on discordant pairs, all systems scored on the same 2,320 queries. The two rows above the rule are the pairs the corpus cannot separate. p-values are uncorrected for the seven comparisons shown; under a Bonferroni correction the Haiku row would not clear 0.05.

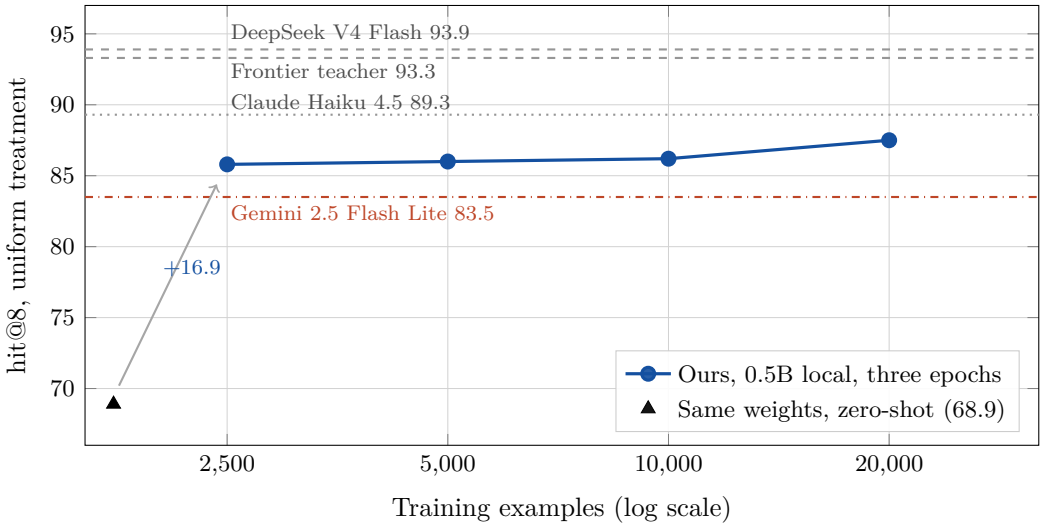

Figure 2: Retrieval quality against training-set size. The first 2,500 examples recover most of the distance from the untrained checkpoint; the next 17,500 recover 1.7 points. The fine-tuned model clears the Gemini 2.5 Flash Lite line and does not reach the top two hosted systems.

This is a narrower and blunter result than the scaling laws reported for domain-specific distillation elsewhere [8], and it is consistent with the general finding that task-specific distillation needs far less data than intuition suggests [11]. Our contribution is the concrete number for this task: budget 2,500 examples, not 25,000.

7 Coverage, format and grounding

Retrieval rank is not the only property that decides whether an enrichment layer is usable. Table 5 reports three others.

Completeness. Every fine-tuned arm returned a sentence for all 2,393 methods. The hosted systems did not: DeepSeek V4 Flash returned 98.0%, Gemini 2.5 Flash Lite 99.0%, Claude Haiku 4.5 99.9%. A missing purpose is not a small error in a knowledge graph, it is a method that cannot be found by any query, and the shortfall compounds with codebase size. At the fourteen thousand

Step	Δ hit@8	Wins	Losses	p
Base → 2,500 (one epoch)	+16.0	483	111	< 0.0001
2,500 one epoch → three epochs	+0.9	146	125	0.224
2,500 → 5,000	+0.2	140	135	0.809
5,000 → 10,000	+0.2	122	118	0.847
10,000 → 20,000	+1.3	136	105	0.053
2,500 → 20,000 (cumulative)	+1.7	159	119	0.019

Table 7: Effect of each training-set step, arms being nested prefixes of one stream. No single step past the first is significant; the cumulative difference across all of them is.

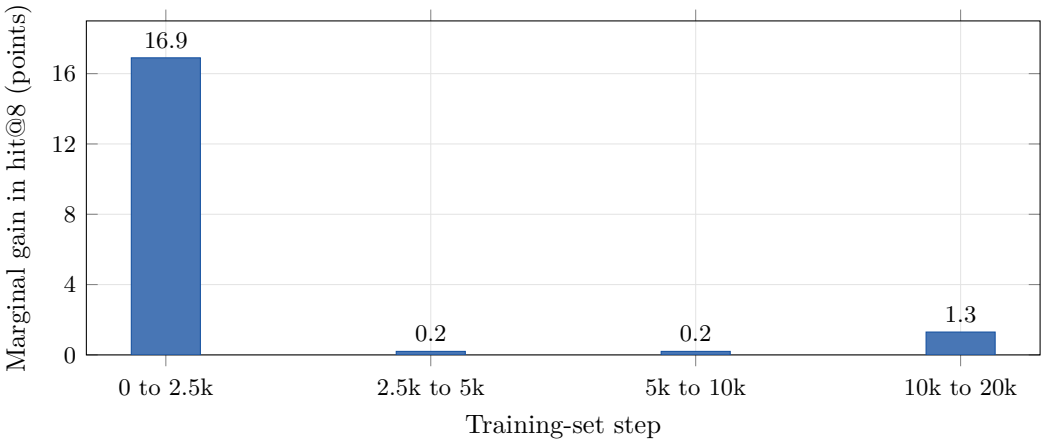

Figure 3: Marginal return per step, each step at least a doubling of the training set. Only the first step is individually significant ($p < 0.0001$); the three that follow are not ($p = 0.809$, 0.847, 0.053). The cumulative 2,500 to 20,000 difference is significant at $p = 0.019$.

methods a real repository carries, a 98% return rate leaves roughly 280 methods with no index entry, and nothing downstream reports that they are absent. The local path has no partial-failure mode of this kind because there is no request to drop: generation either runs or it does not.

Format compliance. 97.7% of the shipping arm's sentences satisfy the length, single-sentence and terminal-punctuation rules, second only to the teacher's 98.6% and ahead of every other hosted system, including DeepSeek V4 Flash at 93.6% and Claude Haiku 4.5 at 89.9%. This is the clearest effect of fine-tuning in the whole study: the same weights untrained comply 87.7% of the time, and 2,500 examples move that to 97.6%. A model that has been shown the output contract obeys it more reliably than a much larger model that has been described it, and the gap does not close with model scale.

Grounding. 0.7% of the shipping arm's sentences name an identifier that does not appear in the method's own source, matching the best hosted rate outside the teacher and below DeepSeek V4 Flash at 1.1%. Fabricated collaborators are the failure mode that makes a generated index actively misleading rather than merely incomplete, and a distilled model at half a billion parameters does not fabricate more than the hosted tier does.

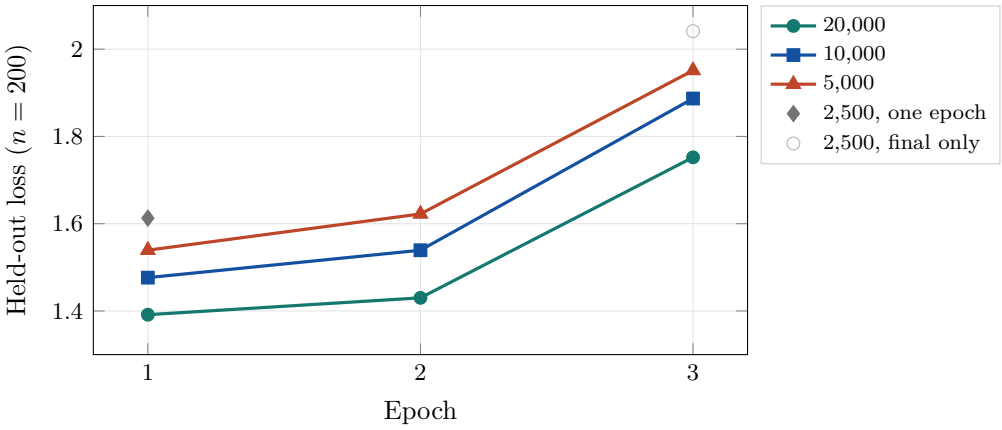

Figure 4: Held-out loss falls with training-set size at every epoch and rises with epochs at every size. Best loss is at epoch one for all three arms with a recorded curve, yet the checkpoints scored in Table 3 are the epoch-three ones. The 2,500-example three-epoch arm retained only its final value. This split is for within-run selection and its language composition differs from training, so the cross-arm reading is indicative only.

Arm	Examples	T4 hours	Loss, epoch 1	Loss, epoch 3	hit@8
2,500 one epoch	2,500	0.10	1.613	n/a	84.9
2,500	2,500	0.31	n/a	2.041	85.8
5,000	5,000	0.61	1.539	1.951	86.0
10,000	10,000	1.29	1.476	1.887	86.2
20,000	20,000	2.43	**1.392**	1.752	87.5
All five arms		4.74 T4-hours			

Table 8: Training cost and held-out loss per arm, on one T4 at fp16 with 10.2 GB peak device memory. GPU-hours are measured; dollar equivalents elsewhere in the paper are modelled at $0.35 per T4-hour. Every scored checkpoint is the final-epoch one.

Calibration. Mean sentence length is 89.4 characters against the teacher's 89.0. The hosted systems scatter either side, DeepSeek V4 Flash at 117.9 and Claude Haiku 4.5 at 75.9. Length uniformity matters under uniform treatment because every passage in the pool competes against every other, and it also governs how much of a context window the enrichment layer consumes when the graph is served to a downstream agent. The distilled model inherits the teacher's calibration closely enough to be substituted without retuning anything downstream of it.

8 Deploying this in-house

The measured price of building the enrichment layer without sending source code anywhere is 5.8 points of hit@8 against the teacher and 6.3 against the leading hosted system. What is bought with those points is categorical, not incremental. No method body, no file path and no identifier leaves the network. The model is a 988 MB artefact that can be pinned, audited and re-run against the same repository revision in five years, which an endpoint under vendor version control cannot be. There is no rate limit, no per-seat metering and no dependency on a vendor's deprecation schedule.

Against Gemini 2.5 Flash Lite, a hosted model KoraGraph runs in its own extraction path, the

in-house model is 4.1 points better at $p < 0.0001$, with no per-call spend after training.

Throughput is not measured here. Our generation runs were taken on a laptop under contention, and a batched server benchmark is the only basis on which we would compare speed.

Three limits bound all of this. One corpus, one embedding model and one fusion configuration produced every figure, so the saturation result may be a property of this corpus and not of the task. The evaluation questions and the training labels came from the same teacher, which would flatter a system whose phrasing matched its own; measured directly, the teacher shares 30.9% of each question's content words against DeepSeek's 36.8%, so it is not the most query-aligned system in the table. And a query here competes against 2,320 methods drawn from 45 unrelated repositories, where a deployed query competes against the methods of one codebase, which are harder distractors.

9 Related work

Knowledge distillation into a smaller student is a long-standing technique [10], and the sequence-level variant, training the student on the teacher's generated text rather than its logits, is what we use here [14]. The most relevant modern result is that task-specific distillation frequently needs far less data than practitioners assume [11], and recent work fits scaling laws to domain-specific compression and finds supervision format to be the dominant lever [8]. Our data-saturation finding is a single point in that space, measured on a retrieval-scored generation task, not a question-answering one, and reported with the divergence between held-out loss and the deployed metric made explicit.

Generating text to attach to a document so that it can be retrieved by queries whose vocabulary it lacks is document expansion [15]. The Purpose line is document expansion for code, with two extra constraints: it is shown to users, and it is required to be grounded in the method source. Semantic code search has an established benchmark tradition [13] whose queries are largely derived from docstrings, which is a different and generally easier signal than the free-form developer questions used here.

Graph-structured retrieval over repositories has been explored as a way to supply models with dependency context rather than whatever a text search returns [21, 17, 22], and a counter-line questions how much of that survives against well-tuned lexical baselines [20, 23]. This paper is orthogonal to that debate. It does not compare graph retrieval against grep; it takes the graph as given and asks who should write the text inside it. Work comparing AST-derived graphs against LLM-extracted knowledge graphs finds the deterministic route more reliable for structure [6], which is the division of labour our system uses: extract structure from the syntax tree, and spend the model only on enrichment. That is exactly the division of labour our system uses, and it is why a 0.5B model is a plausible candidate for the role at all.

On the small-model side, work on retrieval utilisation across scale reports that models at 7B and below often fail to exploit even oracle retrieval [18]. Our result does not contradict it, because our student is not being asked to reason over retrieved context. It is being asked to describe one function it has been handed in full, which is a generation task with a bounded input, and that difference is the most likely reason half a billion parameters suffices here. Work on structural codebase indexes inside coding agents reports a lower cost per solved task for a structural index than for agentic grep, and frames deployment as a question of whether the workload contains changes where structural ranking pays [4].

10 Reproducibility

Every figure in this paper is produced by a scoring harness that reads three inputs: the sealed test corpus, one file of generated sentences per system, and the training metadata written by each run. The harness computes uniform-treatment ranking, Wilson intervals and exact McNemar tests, and emits the tables above without manual transcription. Each number is traced to the file it came from in an accompanying claims ledger, including the two places where a value is absent: the 2,500-example three-epoch arm has no per-epoch loss curve, because its training log was not retained when the compute environment was cleared, so its epoch-one cell is marked unavailable, not interpolated.

Corpus construction is deterministic. Training arms are nested prefixes of one seeded stream, the split assignment is fixed by repository cluster, and every arm file carries a SHA-256 checksum recorded in a manifest alongside the training configuration and its own hash. Given the same corpus and the same seed, the arms rebuild byte for byte.

We are not publishing the corpus or the generated sentence sets at this time.

11 Conclusion

A 494-million-parameter model, fine-tuned for 2.43 T4-hours, under a dollar of commodity GPU time, writes method descriptions that retrieve at 93.8% of the quality of the frontier model that taught it, and beats a hosted model the system was paying for. The remaining gap to the best hosted option is 6.3 points and is statistically real. For a buyer who can use a hosted endpoint, that gap is worth roughly forty-six cents per repository and they should pay it. For a buyer who cannot, this is what the alternative measures at, and it is closer than we expected before running it.

On data, nine tenths of what fine-tuning bought arrived in the first 2,500 examples, while the training objective went on improving for another factor of eight, and anyone optimising this class of task by watching validation loss will spend a great deal of compute after the point where the deployed metric stopped moving.

References

[1] A Middle Path for On-Premises LLM Deployment: Preserving Privacy Without Sacrificing Model Confidentiality. *arXiv preprint arXiv:2410.11182*, 2024.

[2] Anthropic. Claude Haiku 4.5 (`anthropic/claude-haiku-4.5`). https://openrouter.ai/anthropic/claude-haiku-4.5, 2026. Served through the OpenRouter gateway. Accessed August 2026.

[3] Beijing Academy of Artificial Intelligence. BAAI/bge-m3. https://huggingface.co/BAAI/bge-m3, 2024. The embedding model used for all retrieval measurements. Accessed August 2026.

[4] Ishaan Bhola, Adithyan Krishnan, Sravanth Kurmala, and Mukunda NS. Code Isn't Memory: A Structural Codebase Index Inside a Coding Agent. *arXiv preprint arXiv:2606.22417*, 2026.

[5] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-

Knowledge Distillation. In *Findings of the Association for Computational Linguistics (ACL)*, 2024. arXiv:2402.03216.

[6] Manideep Reddy Chinthareddy. Reliable Graph-RAG for Codebases: AST-Derived Graphs vs LLM-Extracted Knowledge Graphs. *arXiv preprint arXiv:2601.08773*, 2026.

[7] DeepSeek. DeepSeek V4 Flash (`deepseek/deepseek-v4-flash-0731`). https://openrouter.ai/deepseek/deepseek-v4-flash-0731, 2026. Served through the OpenRouter gateway. Accessed August 2026.

[8] Lavinia Ghita, Dhruv Desai, and Ioana Boier. Scaling Laws for Task-Specific LLM Distillation. *arXiv preprint arXiv:2606.24747*, 2026.

[9] Google. Gemini 2.5 Flash Lite (`google/gemini-2.5-flash-lite`). https://openrouter.ai/google/gemini-2.5-flash-lite, 2026. Served through the OpenRouter gateway. Accessed August 2026.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015. NIPS 2014 Deep Learning Workshop.

[11] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes. In *Findings of the Association for Computational Linguistics (ACL)*, 2023. arXiv:2305.02301.

[12] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. Qwen2.5-Coder Technical Report. In *arXiv preprint arXiv:2409.12186*, 2024. arXiv:2409.12186.

[13] Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. CodeSearchNet Challenge: Evaluating the State of Semantic Code Search. *arXiv preprint arXiv:1909.09436*, 2019.

[14] Yoon Kim and Alexander M. Rush. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016. arXiv:1606.07947.

[15] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document Expansion by Query Prediction. *arXiv preprint arXiv:1904.08375*, 2019.

[16] NVIDIA. Nemotron 3 Super 120B-A12B (`nvidia/nemotron-3-super-120b-a12b`). https://openrouter.ai/nvidia/nemotron-3-super-120b-a12b, 2026. Served through the OpenRouter gateway. Accessed August 2026.

[17] Siru Ouyang, Wenhao Yu, Kaixin Ma, Zilin Xiao, Zhihan Zhang, Mengzhao Jia, Jiawei Han, Hongming Zhang, and Dong Yu. RepoGraph: Enhancing AI Software Engineering with Repository-level Code Graph. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.14684.

[18] Sanchit Pandey et al. Can Small Language Models Use What They Retrieve? An Empirical Study of Retrieval Utilization Across Model Scale. *arXiv preprint arXiv:2603.11513*, 2026.

[19] Qwen Team, Alibaba Cloud. Qwen2.5-Coder-0.5B-Instruct. https://huggingface.co/Qwen/Qwen2.5-Coder-0.5B-Instruct, 2024. Apache 2.0. The base model fine-tuned in this paper. Accessed August 2026.

[20] Sahil Sen, Akhil Kasturi, Elias Lumer, Anmol Gulati, and Vamse Kumar Subbiah. Is Grep All You Need? How Agent Harnesses Reshape Agentic Search. *arXiv preprint arXiv:2605.15184*, 2026.

[21] Hongyuan Tao, Ying Zhang, Zhenhao Tang, Hongen Peng, Xukun Zhu, Bingchang Liu, Yingguang Yang, Ziyin Zhang, Zhaogui Xu, Haipeng Zhang, Linchao Zhu, Rui Wang, Hang Yu, Jianguo Li, and Peng Di. Code Graph Model (CGM): A Graph-Integrated Large Language Model for Repository-Level Software Engineering Tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.16901.

[22] Martin Vogel, Falk Meyer-Eschenbach, Severin Kohler, Elias Grünewald, and Felix Balzer. Codebase-Memory: Tree-Sitter-Based Knowledge Graphs for LLM Code Exploration via MCP. *arXiv preprint arXiv:2603.27277*, 2026.

[23] Baoyi Wang, Xingliang Wang, Guochang Li, Chen Zhi, Junxiao Han, Xinkui Zhao, Nan Wang, Shuiguang Deng, and Jianwei Yin. Better Call Grep: Evaluating and Improving Grep-Like Lexical Retrieval for Repository-Level Code Completion. In *Proceedings of the 35th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA)*, 2026. arXiv:2601.23254.