

Small Encoder-Decoders as Tool-Calling Backbones: A Capability Decomposition at 270M–1B

Sergey Pletenev

AIRI

alex010rey@gmail.com

Abstract

We present the first capability decomposition of tool calling for sub-1B encoder-decoder language models. We separate format discipline, tool selection, argument fidelity, and multi-turn state tracking across three tiers of difficulty. Recent encoder-decoder research has reported strong general-benchmark results, but it has never tested whether these models can act, and small-model tool-calling research has focused exclusively on decoder-only architectures. To address this gap, we fine-tune T5Gemma 2 (270M/1B) to emit JSON tool calls from a bidirectionally encoded context, “context in the encoder, action out of the decoder”, and train decoder-only twins (Gemma 3, FunctionGemma) on identical data and recipes as controls. At 270M, the encoder-decoder model doubles exact tool selection compared to its decoder-only twin (0.69 vs. 0.34) and more than triples exact calls (0.39 vs. 0.12). It also outperforms the twin on multi-hop question answering, and the gap shrinks but persists at 1B. The negative result is equally clear: every fine-tuned model completes at most 14.5% of the held-out multi-turn dialogues, and failure analysis attributes this drop not to formatting but to wrong tool selection and to under-calling inherited from the training distribution. Code, data and adapters are available at <https://github.com/AlexRey/T5GemmaTools>

1 Introduction

A tool call is a natural language generation problem that has a hard constraint: the model must map an open-ended dialogue context to a JSON object that names a function from a closed set and grounds its arguments in the conversation. For on-device and cost-constrained agents, this generation step must work in models that are far smaller than those usually used for this task. Whether an alternative architecture can do this is unknown.

A wave of concurrent work is re-examining the encoder-decoder transformer at modern scale. This includes the adaptation of decoder-only checkpoints (Zhang et al., 2025b,c), prefix-LM pre-training comparisons (Zhang et al., 2025a), and efficiency-focused small encoder-decoders for edge devices (Elfeki et al., 2025). All of this work evaluates general benchmarks after instruction tuning (Longpre et al., 2023) or distillation, but none of it evaluates tool use in agents. Conversely, small-model tool calling, such as the xLAM family (Liu et al., 2024; Prabhakar et al., 2025) and the Berkeley Function Calling Leaderboard (BFCL) (Patil et al., 2025), has been explored only with decoder-only models. These two lines of work have not yet intersected.

We fine-tune T5Gemma 2 encoder-decoders at 270M and 1B to read the entire context (system prompt, tool schemas, full conversation) in the bidirectional encoder and emit only the action from the decoder. We evaluate on a three-tier difficulty ladder (clean single-turn; messy-schema conversational; fully held-out multi-turn) and control the comparison by training decoder-only twins, Gemma 3 270M/1B (Team, 2025) and the function-calling-specialized FunctionGemma-270M (Google, 2025), on identical data with an identical recipe. Our contributions:

- We present the first tool-calling evaluation of sub-1B encoder-decoder LMs, decomposed into format discipline, tool selection, argument fidelity, and multi-turn state tracking (§4, Figure 1).
- With data and recipe held fixed, the encoder-decoder beats three decoder-only twins at both scales; the margin is largest at 270M (Table 1).
- We report a negative result together with its mechanism: multi-turn dialogue completion collapses for every fine-tuned model ($\leq 14.5\%$). The dominant mechanisms are

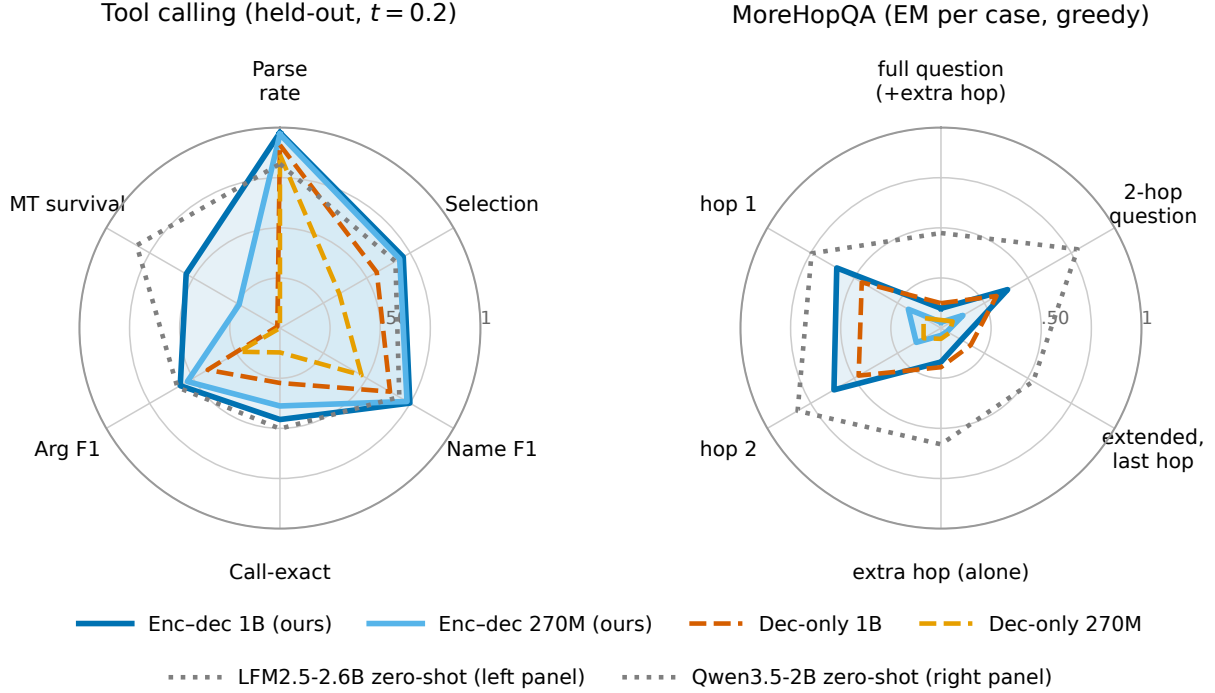


Figure 1: Capability profiles. **Left:** tool-calling decomposition on the held-out single-turn suite (temperature 0.2, 3-seed means) plus multi-turn (MT) survival on the held-out tier-3 dialogues (dialogue-level mean). **Right:** MoreHopQA exact match (EM) per question case (greedy, 1,024-token budget). Solid blue: encoder-decoders (ours); dashed orange: decoder-only twins trained on identical data; dotted grey: the strongest zero-shot baseline of each panel. The encoder-decoder envelope contains its same-recipe twin on every tool-calling axis at both scales.

Model	Parse	Selection	Name F1	Call-exact	Arg F1
<i>Tool-SFT — identical data and recipe</i>					
Enc-dec 270M (ours)	.967 $\pm$.001	.690 $\pm$.009	.732 $\pm$.009	.388 $\pm$.007	.533 $\pm$.008
Dec-only Gemma-3 270M	.865 $\pm$.005	.342 $\pm$.007	.470 $\pm$.005	.121 $\pm$.005	.236 $\pm$.006
Dec-only FunctionGemma 270M	.849 $\pm$.027	.399 $\pm$.012	.519 $\pm$.014	.156 $\pm$.005	.315 $\pm$.006
Enc-dec 1B (ours)	.975 $\pm$.005	.710 $\pm$.009	.748 $\pm$.008	.456 $\pm$.007	.574 $\pm$.005
Dec-only Gemma-3 1B	.913 $\pm$.006	.558 $\pm$.007	.634 $\pm$.004	.274 $\pm$.005	.417 $\pm$.005
<i>Zero-shot</i>					
Dec-only FunctionGemma-270M	.723 $\pm$.004	.284 $\pm$.004	.337 $\pm$.005	.052 $\pm$.001	.091 $\pm$.001
Dec-only Qwen3.5-0.8B	1.00 $\pm$.001	.537 $\pm$.020	.572 $\pm$.021	.289 $\pm$.009	.412 $\pm$.012
Dec-only Qwen3.5-2B	.991 $\pm$.003	.589 $\pm$.012	.621 $\pm$.012	.363 $\pm$.008	.478 $\pm$.009
Dec-only LFM2.5-2.6B	.819 $\pm$.004	.665 $\pm$.007	.687 $\pm$.007	.502 $\pm$.008	.591 $\pm$.008

Table 1: Single-turn capability decomposition (1,000 held-out records; temperature 0.2, mean $\pm$ half-spread over 3 seeds). With data and recipe fixed, the encoder-decoder beats every decoder-only twin at both scales. The suite is in-distribution for the five fine-tuned models; zero-shot rows calibrate difficulty.

wrong-tool selection and a measurable under-calling bias inherited from the training data (§5).

One likely reason for the low adoption of encoder-decoder models in agent and tool-calling systems is the lack of ecosystem support: neither vLLM nor LangChain supports them natively, and neither do the common libraries for supervised fine-tuning (SFT) and reinforcement learning (RL) post-training. For this reason we report quality only, and no time-to-first-token (TTFT) or inference-time

measurements. We hope that papers of this kind renew interest in the architecture.

2 Related Work

Encoder-decoder revival. Zhang et al. (2025a) find that encoder-decoder models match decoder-only models in scaling behavior and fine-tune better. Zhang et al. (2025b) and Zhang et al. (2025c) adapt Gemma checkpoints into encoder-decoders that reach strong general-benchmark scores, and Elfeki et al. (2025) argue that the architecture is more ef-

Model	Selection			Call-exact			Dialogue level (t3)	
	t1	t2	t3	t1	t2	t3	Full succ.	Survival
Enc-dec 270M (ours)	.96	.59	.26	.68	.26	.16	.02	.24
Dec-only Gemma-3 270M	.37	.34	.03	.15	.11	.00	.00	.00
Dec-only FunctionGemma 270M	.48	.37	.03	.24	.12	.00	.00	.00
Enc-dec 1B (ours)	.97	.61	.40	.79	.32	.29	.145	.54
Dec-only Gemma-3 1B	.76	.46	.09	.47	.18	.02	.00	.02
<i>Zero-shot</i>								
Dec-only FunctionGemma-270M	.63	.13	—	.05	.05	—	—	—
Dec-only Qwen3.5-0.8B	.78	.40	—	.46	.20	—	—	—
Dec-only Qwen3.5-2B	.84	.47	.25	.60	.25	.23	.055	.33
Dec-only LFM2.5-2.6B	.94	.56	.58	.79	.38	.52	.305	.83

Table 2: The difficulty ladder (temperature 0.2, seed 0; t1–t3 denote tiers 1–3). Tier-3 columns are per-turn under gold prefixes; full success and survival are dialogue-level over 200 held-out FunReason-MT dialogues. The decoder-only twins fail every dialogue at the first turn.

ficient on-device below 1B parameters. None of these studies evaluates tool use or multi-turn agentic behavior. The fine-tuning advantage reported by Zhang et al. (2025a) makes tool calling a natural test of the architecture, which we run here.

Small-model tool calling. APIGen and APIGen-MT (Liu et al., 2024; Prabhakar et al., 2025) built verified function-calling corpora and the xLAM model family, while BFCL (Patil et al., 2025) is the de facto evaluation standard for multi-turn performance. It reports steep multi-turn degradation at small scale (xLAM-2 at 1B is far below its 3B sibling on the BFCL multi-turn benchmark), but only for decoder-only architectures. We run the same kind of evaluation for encoder-decoders, using a controlled, same-recipe decoder-only comparison rather than leaderboard citations.

3 Experimental Setup

Models. T5Gemma 2 270M and 1B (Zhang et al., 2025c): **encoder-decoder** transformers adapted from Gemma 3 checkpoints, taken from an instruction-tuned (SFT) state and fine-tuned for tool calling (“tool-SFT”; hyperparameters in Appendix B). **Decoder-only** twins Gemma-3-270M/1B-it (the *-it* suffix denotes the instruction-tuned release) and FunctionGemma-270M were fine-tuned on the same data, dialect, and batch regime for the same number of epochs. **Zero-shot** baselines are Qwen3.5-0.8B/2B (Qwen Team, 2026), LFM2.5-2.6B (Amini et al., 2025), and FunctionGemma-270M (Google, 2025), each prompted through its native tool-calling template.

Data. The training data (187,313 examples after transcript-level deduplication) combines **tier 1**,

xlam-function-calling-60k (Liu et al., 2024): single turn, clean schemas (59,614 examples), and **tier 2**, the synthetic `graph_syn` split of ToolMind (Yang et al., 2025): conversational context, messy real-world schemas, and distractor tools (128,699 examples). The held-out evaluation slice (1,000 records: 301 tier-1, 699 tier-2) is split group-aware by conversation (no prefix leakage). **Tier 3** is evaluation-only and fully held out for every model: 200 multi-turn dialogues (490 scored turns) sampled from FunReason-MT (Xu et al., 2025), a BFCL-style dataset from a different construction pipeline and schema dialect.

Format and metrics. The encoder receives the system instruction, one JSON schema per tool, and the full conversation (re-encoded each turn, prior calls replayed). The decoder emits only a JSON call list wrapped in `<tool_call>` markers (template in Appendix A). Decoder-only models receive the same content rendered causally. We report *parse rate* (output yields a well-formed call list), *selection* (exact predicted-tool-set match), *name F1* (per-call tool-name overlap), *call-exact* (name and all arguments exact), and *argument F1* (per-key). Tier-3 scoring is per-turn with gold prefixes (teacher forcing); dialogue-level metrics are *full success* (all turns call-exact) and *survival* (fraction of consecutive correct turns from turn 1).

Decoding and inference. All tool-calling tables use temperature $t=0.2$ (pure sampling; top- k /top- p off), reporting the mean and half-spread over 3 seeds for single-turn suites and seed 0 for tier 3. We deliberately avoid greedy decoding: at bfloat16 (bf16) precision it leads to repetition loops (Appendix C). Inference runs on vLLM (Kwon et al.,

Model	EM	F1	EM per case					
			full+hop	2-hop	ext. last	extra only	hop 2	hop 1
Qwen3.5-2B	65.7	68.4	47.4	78.6	53.1	58.0	82.6	74.5
LFM2.5-2.6B	64.9	66.5	48.8	77.0	55.7	57.7	80.6	69.6
Qwen3.5-0.8B	55.1	58.8	31.5	68.3	35.4	43.7	79.0	72.7
Enc-dec 1B instruct (ours)	33.1	39.6	9.8	38.4	11.6	16.9	61.6	60.0
Gemma-3-1B-it	29.0	36.6	12.4	32.0	17.1	19.5	47.4	45.6
Enc-dec 270M instruct (ours)	9.3	15.2	3.0	12.6	3.5	3.7	14.2	18.8
Gemma-3-270M-it	6.8	15.4	4.0	6.7	4.8	5.4	10.0	10.0

Table 3: MoreHopQA (Schnitzler et al., 2024), zero-shot chain-of-thought (CoT) prompting, greedy decoding, 1,024-token budget: overall exact match (EM) and F1, and per-case EM (1,118 questions per case). Bold = column best.

2023); for the encoder-decoders we use a custom extension that we release with the models and datasets. No prompt is ever truncated, and every run records this.

4 Results

Controlled comparison. Table 1 and Figure 1 (left) show the controlled comparison. At 270M, the encoder-decoder achieves 0.690 exact selection and 0.388 call-exact, while the decoder-only Gemma-3 twin trained on identical data achieves 0.342 and 0.121, respectively. Starting the causal twin from a function-calling-specialized base (FunctionGemma) closes little of the gap (0.399 / 0.156), indicating that the advantage is not due to a weak decoder-only starting point. At 1B, the twin improves to 0.558 / 0.274 but still trails the encoder-decoder (0.710 / 0.456). The gap grows as the scale shrinks, which is the regime most relevant to on-device deployment.

Capability profile at 270M. Format discipline and selection remain intact: parse rate is 0.97 and tier-1 selection is 0.96 at 270M (Table 2). Argument fidelity, however, is the single-turn bottleneck (call-exact 0.39 overall and 0.26 on tier 2). The fine-tuned 270M even exceeds zero-shot LFM2.5-2.6B, a model ten times its size, on selection (0.690 vs. 0.665) while losing on argument exactness (0.388 vs. 0.502); note that tiers 1–2 are in-distribution for the fine-tuned models.

Multi-turn dialogues. The drop from tier 1 to tier 2 is large for every model in Table 2. The difficulty of tier 3, however, lies at the dialogue level. Our best model completes 14.5% of held-out dialogues (survival 0.54), the 270M model completes 2%, and all three decoder-only twins complete none: 197–200 of their 200 dialogues fail at turn

1. Even the strongest zero-shot baseline, LFM2.5-2.6B, fails 70% of dialogues, despite its per-turn tier-3 scores exceeding its tier-2 scores (0.58 vs. 0.56 selection). Dialogues therefore fail because errors compound across turns, not because individual turns are harder; this mirrors the multi-turn degradation of decoder-only models on BFCL (Prabhakar et al., 2025; Patil et al., 2025).

Multi-hop QA. To check that the architectural edge is not specific to the tool-call format, we evaluate the *instruction-tuned* (pre-tool-SFT) checkpoints on MoreHopQA (Table 3; per-case breakdown in Figure 1, right). The cases in MoreHopQA are: *full+hop*, the original 2-hop question plus an extra commonsense/arithmetic/symbolic hop; *2-hop*, the original question; *ext. last*, an extended question over the last hop; *extra only*, the extra hop applied to the 2-hop answer; and *hops 1–2*, the individual sub-questions. The composed question (*full+hop*) is the hardest case for every model. The encoder-decoder wins overall at both scales (33.1 vs. 29.0 EM at 1B; 9.3 vs. 6.8 at 270M); the 1B advantage concentrates on the single-hop sub-questions (+13–14 EM on hops 1–2), with the twin marginally better on the composed variants. 2026-era 2B-class models stay roughly 2× ahead: the architecture helps at matched scale but does not close the gap to a newer, larger model generation.

5 Discussion

Failure taxonomy. We bucket every non-exact single-turn prediction by its first failing stage: unparseable, no call, hallucinated tool, wrong tool, argument keys, argument values. Wrong tool dominates at both scales, accounting for 40–44% of failures, almost entirely on tier 2. Argument values account for ~28%, almost always with selection already correct, and argument keys for ~20%, mostly

optional arguments. Hallucinated tools are essentially absent, with only 0–2 cases per 1,000: both models pick from the offered list even when wrong.

Under-calling. On tier 3 we tested two hypotheses. An *echo* account, in which models re-emit already-executed calls, is refuted: only 4% of selection misses repeat a previous call. A *composition-length* account holds: training targets average 1.37 calls per turn (74% single-call), while tier-3 gold turns average 2.76; the 1B model emits 2.28 and under-calls on 35% of turns. It stops acting too early, consistent with a training signal that contains no stop-vs-continue supervision.

Effect of scale. From 270M to 1B, single-turn selection moves only +0.02, but tier-3 per-turn selection moves +0.14, and full success rises from 0.02 to 0.145. Scale improves state tracking rather than extraction, which is largely solved at 270M.

Benchmark noise. Two audits bound the noise in the benchmarks: 23–24% of tier-2 selection misses pick a near-duplicate *alias* of the gold tool (synthetic tool lists contain duplicates like `Get Car Specifications` vs. `get_car_specs`), and 27–33% of tier-1 argument-value failures require gold values that appear nowhere in the prompt (annotator artifacts). Reported scores are therefore conservative floors.

6 Conclusion

At 270M–1B, encoder-decoder LMs master the extraction side of tool calling and outperform decoder-only twins trained on identical data, most decisively at 270M. Multi-turn state tracking remains a challenge for every architecture tested. Whether bidirectional context encoding can help maintain state, rather than only extract it, remains to be explored; harness-side baselines (constrained decoding, external state summaries) must be established before any architectural claim can be made.

Limitations

The decoder-only twins use the encoder-decoder recipe without any causal-specific tuning, and their training rendered prompts at a 2,048-token budget, truncating 24% of examples. Both choices could understate the twins’ performance. Tiers 1 and 2 are in-distribution for all fine-tuned models; only tier 3 is neutral ground. Tier-3 scoring uses gold-prefix teacher forcing and a single seed. Format adherence of our checkpoints is decoding-sensitive:

greedy decoding collapses the 270M’s tool-JSON via bf16 argmax-tie cascades, while sampling degrades the 1B’s answer-tag discipline on MoreHopQA. Appendix C documents this confound and the decoding policy it motivated.

References

- Alexander Amini, Anna Banaszak, Harold Benoit, Arthur Böök, Tarek Dakhnan, Song Duong, Alfred Eng, Fernando Fernandes, Marc Härkönen, Anne Harrington, Ramin Hasani, Saniya Karwa, Yuri Khrustalev, Maxime Labonne, Mathias Lechner, Valentine Lechner, Simon Lee, Zetian Li, Noel Loo, and 14 others. 2025. [LFM2 technical report](#). Preprint, arXiv:2511.23404.
- Mohamed Elfeki, Rui Liu, and Chad Voegelé. 2025. [Return of the encoder: Maximizing parameter efficiency for SLMs](#). *CoRR*, abs/2501.16273.
- Google. 2025. [FunctionGemma-270M](https://huggingface.co/google/functiongemma-270m-it). <https://huggingface.co/google/functiongemma-270m-it>. Model card, accessed 2026-08.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. [Efficient memory management for large language model serving with PagedAttention](#). In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP ’23)*.
- Zuxin Liu, Thai Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, Rithesh R. N., Liangwei Yang, Silvio Savarese, Juan Carlos Niebles, Huan Wang, Shelby Heinecke, and Caiming Xiong. 2024. [API-Gen: Automated pipeline for generating verifiable and diverse function-calling datasets](#). In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.
- Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. [The Flan collection: Designing data and methods for effective instruction tuning](#). In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.
- Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025. [The Berkeley function calling leaderboard \(BFCL\): from tool use to agentic evaluation of large language models](#). In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19,*

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Manoj Awalganekar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. 2025. [APIGen-MT: Agentic pipeline for multi-turn data generation via simulated agent-human interplay](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*.

Qwen Team. 2026. Qwen3.5 model series. <https://huggingface.co/Qwen>. Model cards for Qwen3.5-0.8B and Qwen3.5-2B, accessed 2026-08.

Julian Schnitzler, Xanh Ho, Jiahao Huang, Florian Boudin, Saku Sugawara, and Akiko Aizawa. 2024. [MoreHopQA: More than multi-hop reasoning](#). *CoRR*, abs/2406.13397.

Gemma Team. 2025. [Gemma 3 technical report](#). *CoRR*, abs/2503.19786.

Zengzhuang Xu, Bingguang Hao, Zechuan Wang, Yuntao Wen, Maolin Wang, Yang Liu, Long Chen, Dong Wang, Yicheng Chen, Cunyin Peng, Chenyi Zhuang, Jinjie Gu, Leilei Gan, Xiangyu Zhao, and Shi Gu. 2025. [FunReason-MT technical report: Overcoming the complexity barrier in multi-turn function calling](#). *CoRR*, abs/2510.24645.

Chen Yang, Ran Le, Yun Xing, Zhenwei An, Zongchao Chen, Wayne Xin Zhao, Yang Song, and Tao Zhang. 2025. [ToolMind technical report: A large-scale, reasoning-enhanced tool-use dataset](#). *CoRR*, abs/2511.15718.

Biao Zhang, Yong Cheng, Siamak Shakeri, Xinyi Wang, Min Ma, and Orhan Firat. 2025a. [Encoder-decoder or decoder-only? Revisiting encoder-decoder large language model](#). *CoRR*, abs/2510.26622.

Biao Zhang, Fedor Moiseev, Joshua Ainslie, Paul Suganthan, Min Ma, Surya Bhupatiraju, Federico Lebrón, Orhan Firat, Armand Joulin, and Zhe Dong. 2025b. [Encoder-decoder Gemma: Improving the quality-efficiency trade-off via adaptation](#). *CoRR*, abs/2504.06225.

Biao Zhang, Paul Suganthan, Gaël Liu, Ilya Philipov, Sahil Dua, Ben Hora, Kat Black, Gus Martins, Omar Sanseviero, Shreya Pathak, Cassidy Hardin, Francesco Visin, Jiageng Zhang, Kathleen Kenealy, Qin Yin, Xiaodan Song, Olivier Lacombe, Armand Joulin, Tris Warkentin, and Adam Roberts. 2025c. [T5Gemma 2: Seeing, reading, and understanding longer](#). *CoRR*, abs/2512.14856.

A Prompt Template

The encoder input renders the record as plain text; the decoder target is the call list alone. Abbreviated example (tier-2 record):

```
[system] You are a function-calling
assistant. Given the available tools
and the conversation, output the tool
calls that resolve the user's request
as a JSON list inside
<tool_call>...</tool_call>. If no tool
applies, output an empty list. Do not
invent tools or arguments.
[tools]
{"name":"Earnings History",
 "description":"Retrieve the earnings
history for a specific stock.",
 "parameters":{...}}
{"name":"tickers_list", ...}
... one JSON schema per line ...
[conversation]
user: ...
assistant: <tool_call>[...]</tool_call>
tool: {...}
assistant: ...
user: Let's start with AAPL. Could you
retrieve its latest earnings history
and ...?
assistant:
```

Target:

```
<tool_call>[{"name":"Earnings History",
 "arguments":{"ticker":"AAPL"}}, ...]
</tool_call>
```

Prior tool calls are replayed in the same dialect inside the conversation; the full context is re-encoded at every turn. Decoder-only twins receive the identical content through a causal chat template. Tool-call markers are plain text (no added vocabulary): resizing embeddings desynchronizes the encoder and decoder vocabularies in this model family.

B Hyperparameters

	270M	1B
Learning rate	5×10^{-6}	3×10^{-6}
Epochs	2	2
Effective batch	64	64
Max source / target len	2048 / 512	2048 / 512
Seed	17	17

Table 4: Tool-SFT hyperparameters (identical for the decoder-only twins, with the 270M learning rate). Low learning rates reflect fine-tuning already-SFT'd checkpoints. Training used $2 \times A100/H200$ -class GPUs (270M ~hours) and a $4 \times A100$ node (1B).

Evaluation generates up to 512 new tokens (1,024 for models that emit reasoning traces before calls, and for all of MoreHopQA), with prompt budgets

of 16,384 tokens (single-turn) and 8,192–16,384 (tier 3), verified untruncated for every reported run.

C Greedy Decoding at bf16

Greedy decoding is tie-brittle at bf16: 3.4% of the 270M model’s greedy steps (0.9% at 1B) are exact bf16 argmax ties, resolved differently by different kernel orders. Teacher-forced log-probabilities agree across engines to $|\Delta| \approx 10^{-4}$, yet free-running greedy decoding diverges at ties and the 270M then collapses into repetition loops (parse rate 0.96 on one engine vs. 0.47 on another, on identical records). Temperature-0.2 sampling makes tie resolution random and escapes the loops: it reproduces greedy-level scores within seed noise (± 1 pp) on both engines, which motivated the decoding policy of §3. The reverse sensitivity also exists on MoreHopQA: the same sampling degrades the 1B’s `<answer>`-tag discipline (unclosed tags on 19–37% of generations, depending on the seed), so we decode that benchmark greedily, with one decoding configuration per table, chosen by what the metric is sensitive to.