# Training Tab Namer 6.9M from Scratch

A comparison with FLAN-T5, PorkiCoder's installed tab namer

PorkiCoder Research · Final experiment report · 2026-11-25 18:45 UTC

## Final test accuracy

Tab Namer 6.9M is the model evaluated in Round 2. It has six layers and 6.9 million parameters. We gave it and FLAN-T5 the same 1,200 inputs and asked two judges how useful each title would be on a terminal tab.

| Judge | Tab Namer 6.9M | FLAN-T5 | Gain |
|---|---|---|---|
| Gemma 4 | 73.50% | 67.25% | +6.25 percentage points |
| Flash-Lite | 76.25% | 72.42% | +3.83 percentage points |

Judge-rated naming accuracy is the share of all 1,200 inputs whose title earned at least 6/10 for usefulness. Every input counts once, including requests that returned no title. Several titles can suit a task. See 30 pairs of actual titles in Appendix A.

## Abstract

We trained Tab Namer 6.9M on private conversation transcripts and 46,423 teacher-written titles. It has 91.0% fewer parameters than FLAN-T5. On inputs reconstructed from 95 held-out sessions, Gemma 4 rated 882 titles from Tab Namer 6.9M and 807 from FLAN-T5 useful. Gemma 4's mean score favored Tab Namer 6.9M by 0.317 points on a ten-point scale (95% interval +0.159 to +0.486). Flash-Lite measured a gain of 0.194 points (+0.068 to +0.332). Tab Namer 6.9M met the criteria we set before judging.

Most of the improvement came from naming assistant updates. Those accounted for 67 of the 75 additional titles Gemma 4 rated useful. User requests accounted for the remaining 8. Flash-Lite also found its largest gain on assistant updates. Its mean scores on user requests were nearly identical.

An earlier version, called the Round 1 model in this paper, missed the primary criterion on a separate 680-input test. For Round 2 we reserved fresh sessions, trained Tab Namer 6.9M from random weights and chose its settings using development data. Human preference and the minimum size needed for this task remain open questions.

## 1. Question and task

A tab title should help its owner recognize the current task among several open terminals. Can a model small enough to run locally do that as well as the deployed model? Training targets contain two or three words and at most 32 characters. The application accepts one to three sanitized words. We froze the installed FLAN-T5-derived model [1] with its JavaScript preprocessing, tokenizer, ONNX graphs and decoder.

Tab Namer 6.9M learned its vocabulary and weights from scratch. DeepSeek supplied examples of good titles. A decoding rule then limited each title to eligible words from the input. The final test evaluates that combination.

**How FLAN-T5 learned the task.** Google's FLAN-T5-small already knew patterns of English and had practiced following instructions. PorkiCoder adapted those weights by showing it terminal text paired with short titles. In Session 10, title fine-tuning raised the useful-title rate from 15.8% for Google's original weights to 87.8% for the title-tuned model on the same 500 development inputs. Session 10 comparison [6].

Session 11 continued from those title-tuned weights for four more passes through a filtered mix of existing titles and examples labeled by Grok, Opus and Codex. The learning rate was 0.0001 and the examples had equal training weight. That checkpoint, flat1e4, became the installed FLAN-T5. It generates one token at a time by choosing the most likely next token, with no rule requiring words from the input. Its 85.7% and 84.5% accuracy figures came from two earlier 1,000-input tests. The final accuracy table gives both systems the same November inputs. Session 11 training record [7] and released model card [8].

## 2. Reconstructing real naming inputs

PorkiCoder uses short titles for user requests and for updates written by an assistant. We could reconstruct the text presented to the namer from one operator's personal and work laptop backups. The backups do not record every historical naming attempt, its timing or its outcome. We kept the first visible paragraph, normalized whitespace and control characters, capped it at 400 Unicode characters, and supplied the agent name and project basename. We excluded tool output, hidden reasoning and partial-message variants.

DeepSeek V4.1 Flash 1017 Fast wrote all 50,000 accepted training titles with reasoning disabled. The initial labeling pool contained 56,183 distinct cleaned messages. Of these, 2,766 received no accepted label and 3,417 remained unreviewed. Round 2 used 46,423 of the accepted pairs after reserving sessions for evaluation.

| Round 2 partition | Examples | Role |
|---|---|---|
| Retained teacher labels | 46423 | Training only |
| User requests / assistant updates | 6,087 / 40,336 | Natural source imbalance |
| Development | 738 in 70 sessions | Selection and diagnostics |
| New final test | 1200 in 95 sessions | 600 requests and 600 updates |
| Round 1 final test | 680 in 56 sessions | Retired after Round 1 |

A deterministic hash order reserves 261 formerly training sessions before the second round starts. From those sessions, fixed hash rankings select 600 examples of each source kind for final evaluation. Every message in all reserved sessions is excluded from fitting, including messages not selected into the test. This removes 3,577 previously available teacher rows. The remaining label lines are reused byte-for-byte, with no new labeling calls.

Earlier models had seen some sessions that we later reserved for Round 2. We therefore rebuilt the tokenizer and started the weights from scratch. We also excluded matching normalized openings from the language corpus. The split protects against reuse of a session. Different sessions can still discuss the same project.

## 3. Training and decoding

**First, learn from ordinary text.** Tab Namer 6.9M began with random weights. Imagine a transcript beginning “Compare database query”. The model assigns probabilities to what comes next. When it sees the actual next piece of text, the training program adjusts its weights to make that continuation more likely next time. Every position in a transcript provides a prediction to practice. At this stage there is no title to guess.

**Then, learn what belongs on a tab.** We gave the model a message followed by its DeepSeek title. It now received training feedback for predicting the title. A message can mention a database, indexing and latency. A useful title has room for only a few of those ideas. The teacher examples show which words tend to distinguish one task from another. Both this stage and FLAN-T5 fine-tuning use title supervision. The difference is what each model knew before that stage began.

| Training choice | Tab Namer 6.9M | FLAN-T5 |
|---|---|---|
| Starting point | Random weights and a tokenizer learned on this corpus | Google’s instruction-tuned FLAN-T5-small weights and tokenizer |
| Local training signal | Predict transcript tokens, then predict DeepSeek titles for 46,423 inputs | Continue title training on the earlier free-form, multi-teacher title mix |
| How a title is written | Compare up to four continuations at a time, within titles built from eligible source words | Greedy generation with the installed title sanitizer |

Tab Namer 6.9M uses a GPT-2-style causal decoder [2] with six layers, width 256, four attention heads, context length 256 and 8,192 byte-level BPE tokens. Sharing the input and output embeddings gives 6,901,760 parameters. On the PGX’s NVIDIA GB10 we used AdamW with a peak learning rate of 6e-4, betas of 0.9 and 0.95, weight decay and dropout of 0.1, gradient clipping at 1.0, 100 warmup updates and cosine decay. The master weights used FP32 and matrix operations used BF16.

The new language corpus contains 49,344 documents and 5,566,631 packed tokens. Four thousand updates of 64 sequences of 256 tokens yield 65,536,000 token exposures; repetition does not increase unique data. The trainer reported 590.6 seconds for this stage, excluding preparation. There is no matched title-only-from-random arm, so the benefit or necessity of language pretraining is not isolated.

For title training we provided up to 190 input tokens followed by a delimiter and the teacher’s title. Only errors on the title tokens and its end marker contributed to the loss. Tab Namer 6.9M used six passes over the labeled data, or 4,353 updates at batch size 64 with a peak learning rate of 3e-4. We also tried 2,177 additional updates at 2e-4 after 50 warmup steps. One variant shuffled the training set. Another used 32 user requests and 32 assistant updates in each batch.

Round 1 also tested reward training with a local judge and REINFORCE leave-one-out updates [3]. At 50,000 labels, the reward-trained checkpoint led its supervised counterpart by 0.01 points on development data (95% interval −0.215 to +0.225). Round 2 therefore focused on supervised title training and decoding. Inkling scored the development comparisons. Gemma 4 and Flash-Lite scored the final tests.

On the same 100 development inputs, the Round 1 training runs with 6k, 8k, 20k and 50k labels produced 45, 39, 63 and 72 useful titles. Inkling rated 70, 70, 68 and 59 FLAN-T5 titles useful in the corresponding comparisons, even though FLAN-T5 returned the same titles each time. The shifting ratings limit comparisons across those runs.

| Decoder | Permitted titles |
|---|---|
| Unrestricted four-beam | Normal model generation followed by the installed title sanitizer |
| Source words 2 | Any ordered pair of distinct eligible visible source words |
| Source words 2–3 | All preceding pairs, plus permutations of adjacent triples of eligible source words |

Source-constrained decoding uses four beams, a whole-title byte-BPE prefix trie, length penalty 1.0, and at most 24 new tokens including EOS. It considers at most 96 distinct visible words, with a fixed lexical stoplist and capitalization variants. Titles must round-trip through tokenization and satisfy the 32-character runtime limit. A one-word fallback is allowed only when no pair is available. Three-word candidates come from local windows of eligible words, avoiding an unrestricted cubic combination search. These constraints add no trained parameters.

## 4. Choosing a decoder and model size

Inkling scored anonymized titles from 1 to 10. A score of at least 6 counted as useful, and a score of 2 or lower counted as unusable. All judges saw the task paragraph and proposed titles. Only the naming models also saw the agent name and project. A judge therefore had less context than the operator choosing among open tabs.

We first compared settings on 100 development inputs, evenly split between user requests and assistant updates. The best mean-scoring setting for each decoder and the leader by useful-title count advanced to all 738 inputs. We required an advantage over FLAN-T5 both on the full set and on the 638 inputs outside the initial screen. The table gives ordinary averages. The paired analysis gives equal weight to requests and updates.

**Inkling scores on all 738 development inputs.**

| Configuration | Mean / 10 | Judge-rated accuracy (useful ≥ 6) | Unusable ≤ 2 |
|---|---|---|---|
| Natural recovery / source words 2–3 | 7.111 | 85.64% (632/738) | 0 |
| FLAN-T5 | 6.194 | 65.31% (482/738) | 31 |
| Six epochs / unrestricted beams | 6.251 | 63.69% (470/738) | 33 |
| Tab Namer 6.9M | 7.209 | 87.40% (645/738) | 0 |
| Six epochs / source words 2 | 6.793 | 81.30% (600/738) | 1 |

Tab Namer 6.9M used six epochs of title training and a decoder that permits two or three source words. It met both development criteria.

Changing only how we decoded the same trained weights raised the number of useful titles from 470 with unrestricted generation to 600 with two source words and 645 when three-word titles were also allowed. That comparison is the clearest evidence for a specific contributor to the gain.

**Why the result favors Tab Namer 6.9M.** On those same 738 development inputs, FLAN-T5 produced 482 useful titles. With unrestricted decoding, the Tab Namer 6.9M weights produced fewer useful titles than FLAN-T5. Allowing two or three words drawn from the message moved them ahead. This rule makes it easier to retain a distinctive term such as “latency”, although it can yield awkward word order. The 40,336 assistant updates in the title-training set also make familiarity with that style

a plausible explanation for the larger gain on updates. We did not vary the training mix in isolation to measure that effect.

The final test establishes which complete system performed better on these inputs. To measure the contribution of pretraining itself, we would need to vary the starting weights while holding the architecture, examples, title training and decoder fixed. To test whether FLAN-T5 could close the gap, we would also train it on the same DeepSeek labels and compare both decoding rules.

We then asked how much smaller the model could be. Before training these variants, we expanded the comparison from three depths to all six. We kept the layers listed below from Tab Namer 6.9M, then trained each variant for 2,178 updates at batch size 64, using natural sampling, a peak learning rate of 2e-4 and 50 warmup updates. We compared checkpoints at steps 1,089 and 2,178 with the original Tab Namer 6.9M.

| Layers | Parameters | Parent layers retained |
|---|---|---|
| 6 | 6,901,760 | All six |
| 5 | 6,112,000 | 0, 1, 2, 4, 5 |
| 4 | 5,322,240 | 0, 2, 4, 5 |
| 3 | 4,532,480 | 0, 2, 5 |
| 2 | 3,742,720 | 0, 5 |
| 1 | 2,952,960 | 2 |

All six depths completed training. On the 100-input screen, Inkling marked 87 titles from Tab Namer 6.9M useful, compared with 86 from the best five-layer variant, 84 from the best three-layer variant and 65 from FLAN-T5. We paused the broader size study after 392 of 738 inputs. We chose Tab Namer 6.9M for the final test on the strength of its completed development evaluation. The minimum viable depth remains open.

## 5. Two independent final tests

Before final scoring, we froze the weights, tokenizer, preprocessing, decoder and both titles for each input. Gemma 4 (31B, high thinking requested) was the primary judge [4]. Gemini 3.5 Flash-Lite, called Flash-Lite here, provided continuity with earlier studies. Each judge saw titles in a fixed anonymous order for each input. Identical titles received identical scores.

To pass, a model had to beat FLAN-T5 on mean score with a positive lower 95% confidence bound, produce at least as many useful titles and at most as many unusable titles, and match or exceed the FLAN-T5 mean for each input type. We estimated uncertainty by resampling whole sessions 10,000 times. We also specified that a half-precision copy could replace the original weights only if every title stayed identical on development and final inputs.

**Round 1 model versus FLAN-T5 (680 inputs).**

| Judge | Round 1 model accuracy | FLAN-T5 accuracy | Paired mean gain [95% interval] |
|---|---|---|---|
| Gemma 4 | 78.24% (532/680) | 73.97% (503/680) | +0.105 [-0.112, +0.295] |
| Flash-Lite | 82.50% (561/680) | 75.59% (514/680) | +0.245 [+0.051, +0.411] |

The Round 1 model earned higher overall means and more useful titles than FLAN-T5 under both judges. It missed two primary criteria. Gemma 4's interval included zero, and user requests averaged 0.076 points lower than FLAN-T5. We retired these 680 inputs from subsequent training and testing.

**Tab Namer 6.9M versus FLAN-T5 (1,200 inputs).**

### Gemma 4

| Configuration | Mean / 10 | Judge-rated accuracy (useful ≥ 6) | Unusable ≤ 2 |
|---|---|---|---|
| Tab Namer 6.9M | 6.799 | 73.50% (882/1,200) | 29 |
| FLAN-T5 | 6.482 | 67.25% (807/1,200) | 67 |

The paired mean difference was +0.317 with a session-bootstrap 95% interval of [+0.159, +0.486]. Tab Namer 6.9M won 544 of the 1,200 head-to-head comparisons (45.33%), tied 146 (12.17%) and lost 510 (42.50%).

| Gemma 4 by input type | Tab Namer 6.9M accuracy | FLAN-T5 accuracy | Tab Namer 6.9M mean | FLAN-T5 mean |
|---|---|---|---|---|
| Assistant update | 80.67% (484/600) | 69.50% (417/600) | 7.128 | 6.569 |
| User request | 66.33% (398/600) | 65.00% (390/600) | 6.471 | 6.394 |

### Flash-Lite

| Configuration | Mean / 10 | Judge-rated accuracy (useful ≥ 6) | Unusable ≤ 2 |
|---|---|---|---|
| Tab Namer 6.9M | 6.851 | 76.25% (915/1,200) | 24 |
| FLAN-T5 | 6.657 | 72.42% (869/1,200) | 49 |

The paired mean difference was +0.194 with a session-bootstrap 95% interval of [+0.068, +0.332]. Tab Namer 6.9M won 528 of the 1,200 head-to-head comparisons (44.00%), tied 124 (10.33%) and lost 548 (45.67%).

| Flash-Lite by input type | Tab Namer 6.9M accuracy | FLAN-T5 accuracy | Tab Namer 6.9M mean | FLAN-T5 mean |
|---|---|---|---|---|
| Assistant update | 86.00% (516/600) | 77.67% (466/600) | 7.336 | 6.952 |
| User request | 66.50% (399/600) | 67.17% (403/600) | 6.367 | 6.362 |

On individual inputs, both judges preferred Tab Namer 6.9M on 406 inputs and FLAN-T5 on 389. They both recorded a tie on 96. Their win, tie or loss classifications matched on 891/1,200 inputs. They preferred opposite systems on 231, and exactly one judge gave a tie on 78.

Flash-Lite preferred FLAN-T5 on more individual inputs, yet gave Tab Namer 6.9M the higher mean. Counting wins gives a narrow win the same weight as a decisive one. Averaging scores preserves the margin. On user requests, the two were close. Flash-Lite marked 399 titles from Tab Namer 6.9M useful and 403 from FLAN-T5.

Tab Namer 6.9M returned a title for 1,200 inputs and FLAN-T5 returned one for 1,173. Their titles matched exactly on 52 inputs. Could missing FLAN-T5 titles explain the overall difference? Restricting the comparison to the 1,173 inputs where both systems returned a title gave mean

differences of +0.194 for Gemma 4 and +0.073 for Flash-Lite, weighting both input types equally. This descriptive check was added after prediction and has no role in the primary decision.

Tab Namer 6.9M met all four prespecified criteria under Gemma 4.

We reconstructed every paired score from the saved API responses, verified the anonymous candidate assignments, and matched all 1,200 inputs exactly once for each judge. The audit covered 100 accepted Gemma 4 responses and 100 accepted Flash-Lite responses.

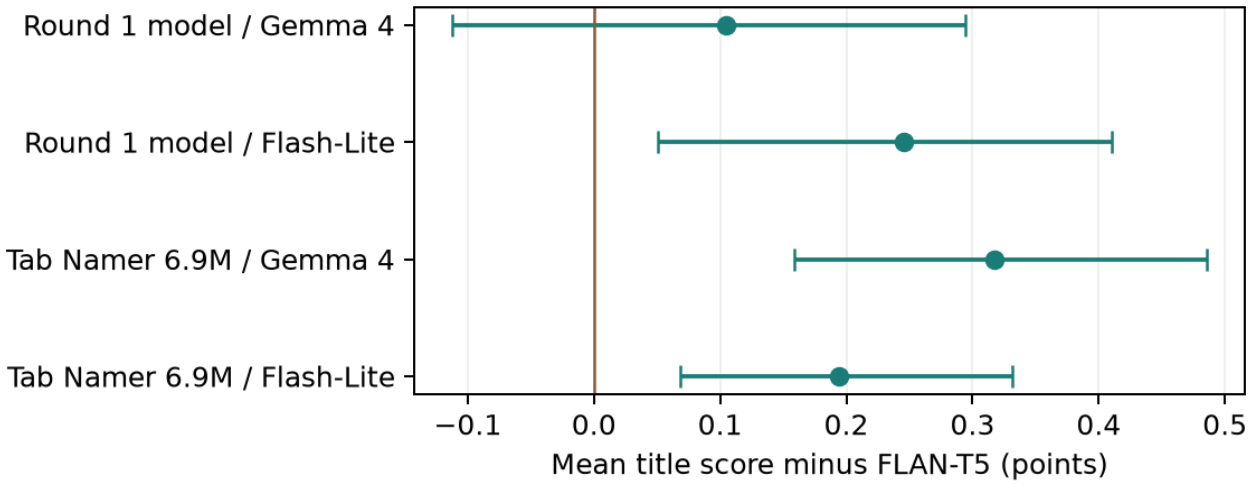



Figure 1. Each model is compared with FLAN-T5 on the same inputs within its own round. Bars show 95% intervals from resampling sessions. The rounds used different inputs, and these intervals do not capture judge bias.

An index collision in a development judging request was corrected before any responses were accepted. The final parser accepts one JSON object. The protocol permits splitting a repeatedly timed-out request while retaining every case and saving each response. The audit above verifies the actual scores used in the analysis.

## 6. Footprint, runtime, and reproducibility

| Measurement | Tab Namer 6.9M | FLAN-T5 |
|---|---|---|
| Parameters | 6,901,760 | 76,961,152 |
| Weight files | 27,614,528 bytes | 248,926,900 bytes in three ONNX graphs |
| Warm median | 30.24 ms | 14.32 ms |
| Warm 95th percentile | 80.23 ms | 21.52 ms |

With FP16 storage, Tab Namer 6.9M reproduced 738/738 development titles exactly. It advanced to the predeclared final identity check.

On the final inputs, half-precision storage reproduced 1,199/1,200 full-precision titles. The reported storage format is FP32. The weights are loaded into FP32 for inference.

The 27.6 MB Tab Namer 6.9M weights occupy about one ninth of the space of the FLAN-T5 graphs. On the laptop, Tab Namer 6.9M took 2.1 times as long at the median. Both systems produced their titles sequentially on the same 1,200 inputs. Reported weight sizes exclude dependencies.

We timed one request at a time on an Apple M4 Max with 16 CPU cores, 48 GiB of memory and macOS 15.7.4. Warm timings exclude the first request. Tab Namer 6.9M used four PyTorch CPU threads and FP32 computation. FLAN-T5 used the installed INT8 ONNX runtime with one intra-operation and one inter-operation thread. Its three graphs duplicate some tensors. These measurements describe the two implementations an operator would actually run.

PorkiCoder can serialize its prepared paragraph directly for the model. The packaged reader preserved all 1,200 of 1,200 evaluated titles when given those exact strings. This path preserves any space at the 400-character cutoff. The separate plain-text reader applies paragraph normalization to new text.

**Six examples fixed before selection.** Both systems generated these titles for the same synthetic inputs. Appendix A gives a longer side-by-side comparison.

| Synthetic input | Tab Namer 6.9M | FLAN-T5 |
|---|---|---|
| Fix the login redirect loop after sessions expire. | Login Loop Redirect | Fix Login Redirect |
| I'm adding keyboard shortcuts to the image preview. | Keyboard Preview | Add Keybar shortcuts |
| Compare database query latency before and after indexing. | Database Query Latency | Compare Latency |
| I'm restoring file search when an external drive reconnects. | File Search Restoring | Restore File Search |
| It broke again. Can you fix it? | Broke Fix | Fix Broken Recall |
| The parser now rejects malformed dates and reports the offending field. | Malformed Dates | Restore Offending Field |

“Database Query Latency” preserves the object being measured. “Keyboard Preview” drops the shortcuts. “Restore File Search” reads more naturally than “File Search Restoring.” The vague repair request gives both models little to work with.

The full 50k label corpus has a recorded usage-based cost estimate of $5.090627; the second round reuses its retained labels without new teacher calls. One Gemini 4 Neon recipe consultation was estimated at $0.092970. First-round Flash-Lite final evaluation was estimated at $0.075464. Second-round Flash-Lite is estimated at $0.131791. Flash-Lite estimates use published token rates [5]. Hosted Gemma billing is not measured; a missing price or a timed-out request without usage must not be reported as zero cost. The owned PGX incurs no cloud rental charge; electricity and depreciation are excluded.

We verified the privately recovered checkpoints by SHA-256. The Tab Namer 6.9M package includes its weights, tokenizer, inference code, dependency versions and aggregate evaluation. It runs locally on CPU. FLAN-T5 remains installed in PorkiCoder. The package contains no conversation corpus or per-input judge files.

Second-round input pool: ff35a99db8b81fc34e9326503933090a
f6eb67f62bd6008d2e475bf027fa9405

Retained teacher labels: 4dfe2591ba9450489973a613cb837e40
1dd1622b9f049068c699b221e0e70fc9

Fresh language weights: 03465baed4e5e30e0323109dd0b1a446
d4605d7e3b37b596474695bec190aded

Fresh tokenizer: bcab2eed9f4386c9889fb8ad2a93dc65
f3b6d50eb3edfe83309342ec72e214b0

Final protocol: a337d474aabafd0a1b2db8adb7c4222c
08535a801ea0912d7ef6a9a8bbf6720b

Tab Namer 6.9M weights: d6449e56c0ade23a8c0e6a946cffd8de
8ce1b0fa3e2c298a5970394f0de834eb

Final comparison freeze: 435ebdbb544c349e59d961f2144b0cd8
b67969c7f1b50cdf8cbba3f95943dd13

## 7. Limitations

All sessions came from one operator. Reconstructed paragraphs can differ from live terminal text, and the same project can appear in several sessions. Teacher filtering and the large share of assistant updates in training can affect the result. Both judges can share biases. We used one seed per recipe, reused development inputs across choices and made no multiple-testing adjustment across the two final tests. We did not measure human preference or isolate the effect of language pretraining. The unfinished depth study leaves open whether fewer parameters would suffice.

## 8. Future experiments

**Does the operator recognize the right tab?** Present title pairs from fresh sessions in random order, hide the model identity, and let the operator choose either title or a tie. Include both requests and assistant updates. On a subset, show the surrounding tabs and time how long it takes to find the intended one. This tests the practical meaning of a higher judge score.

**How many layers are needed?** The paused size comparison has 346 development inputs left. A future study can complete it, select the smallest depth that matches Tab Namer 6.9M under the existing rule, then compare the frozen finalist with Tab Namer 6.9M on fresh held-out sessions. The current final test has already informed this report.

**What did language pretraining buy?** Start matched models from the same random initialization and tokenizer. Give one the recorded language-pretraining phase and train the other directly on titles. Hold the labels, title updates and decoder fixed, report the extra compute, and repeat across seeds. This would separate the contribution of language pretraining from the gain already observed from source-word decoding.

## References


[1] Chung et al. (2022). Scaling Instruction-Finetuned Language Models. https://arxiv.org/abs/2210.11416

[2] Hugging Face. Transformers GPT-2 implementation and documentation.
https://huggingface.co/docs/transformers/model_doc/gpt2

[3] Ahmadian et al. (2024). Back to Basics: Revisiting REINFORCE-Style Optimization for Learning from Human Feedback in LLMs. https://aclanthology.org/2024.acl-long.662/

[4] Google DeepMind. Run Gemma with the Gemini API. https://gemma.ai/deployment/gemini-api/

[5] Google. Gemini Developer API pricing. https://ai.google.dev/gemini-api/docs/pricing

[6] PorkiCoder Research (2026, August 17). Session 10. Title fine-tuning, quality and latency.
https://porkicoder.com/research/session10_results.html

[7] PorkiCoder Research (2026, August 17). Session 11. Continued FLAN-T5 fine-tuning on the August 12 title mix.
https://porkicoder.com/research/session11_results.html

[8] PorkiCoder. PorkiCoder Tab Namer 77M. Released model card. https://huggingface.co/porkr/porkicoder-tab-namer-77m

## Appendix A. Thirty side-by-side title comparisons

We ran both evaluated systems on 30 synthetic messages written for illustration: 15 user requests and 15 assistant updates. The first six were fixed before model selection. The other 24 were written after the test. Both systems received the same message, agent and project on each row. The table includes every output in input order. These examples carry no judge scores and do not contribute to the accuracy figures.

| Illustrative input | FLAN-T5 | Tab Namer 6.9M |
|---|---|---|
| Fix the login redirect loop after sessions expire. | Fix Login Redirect | Login Loop Redirect |
| I'm adding keyboard shortcuts to the image preview. | Add Keybar shortcuts | Keyboard Preview |
| Compare database query latency before and after indexing. | Compare Latency | Database Query Latency |
| I'm restoring file search when an external drive reconnects. | Restore File Search | File Search Restoring |
| It broke again. Can you fix it? | Fix Broken Recall | Broke Fix |
| The parser now rejects malformed dates and reports the offending field. | Restore Offending Field | Malformed Dates |
| Keep the selected row visible when the table refreshes. | Refresh Table Row | Table Row |
| The timer now pauses while the settings window is open. | Restart Settings Window | Settings Timer |
| Why does the chart repeat Sunday at the end of the week? | Chart Repeats | Chart Sunday |
| I traced the blank thumbnail to an empty color profile. | Trace Color Blurt | Thumbnail Profile |
| Add a confirmation before clearing saved filters. | Add Confirmation | Clearing Confirmation |
| I found the missing tax line in the invoice preview. | Missing Invoice Line | Invoice Preview Line |
| Where did the second copy of the calendar event come from? | [no title] | Calendar Event |
| The export now quotes cells that contain a comma. | Export Comma Export | Export Cells |
| Show me which tests still rely on the old clock. | Show Old Clock | Show Tests |
| I changed the search box to ignore accents. | Missing Search Box | Search Box Changed |
| Keep unfinished drafts when I switch notebooks. | Keep Paper Rewrites | Drafts Switch |
| The progress bar was measuring bytes before decompression. | Progress Bar Recorded | Progress Bar Measuring |
| Make the map legend readable on a small screen. | Map Legend | Map Legend |
| The date selector now keeps leap day when changing views. | Date Selector leaps | Day Views |
| Can we sort by file size without moving folders? | Sort File Size | File Size |
| I moved the tooltip so it no longer hides the axis labels. | Fix Tooltip Labels | Tooltip Labels |
| Find the first version that stopped accepting pasted images. | Find First Version | Stopped First Version |
| The settings panel now restores the previous font size. | Restore Font Size | Settings Panel |

| Illustrative input | FLAN-T5 | Tab Namer 6.9M |
|---|---|---|
| Can you show the difference between these two color palettes? | Color palettes Comparison | Palettes Difference |
| I fixed the receipt printer margin on narrow paper. | Fix Printer Margin | Printer Receipt Margin |
| Why are some paragraphs missing from the print preview? | Fix Print Preview | Print Preview |
| The importer now reports the first malformed CSV row. | Malformed CSV Row | Row Malformed CSV |
| Let me rename a playlist while the music keeps playing. | Rename playlist | Playlist Rename |
| I added a test for undo after dragging a card. | Test Undo | Test Dragging |