

# Micro Language Models Enable Instant Responses

Wen Cheng\* Tuochoa Chen\* Karim Helwani<sup>†</sup> Sriram Srinivasan<sup>†</sup>  
Luke Zettlemoyer\* Shyamnath Gollakota\*

\*Paul G. Allen School of Computer Science & Engineering, University of Washington

<sup>†</sup>Meta AI

## Abstract

Edge devices such as smartwatches and smart glasses cannot continuously run even the smallest 100M–1B parameter language models due to power and compute constraints, yet cloud inference introduces multi-second latencies that break the illusion of a responsive assistant. We introduce micro language models ( $\mu$ LMs): ultra-compact models (8M–30M parameters) that instantly generate the first 4–8 words of a contextually grounded response on-device, while a cloud model completes it; thus, masking the cloud generation latency. We show that useful language generation survives at this extreme scale with our models matching several 70M–256M-class existing models. We design a collaborative generation framework that reframes the cloud model as a continuator rather than a respondent, achieving seamless mid-sentence handoffs and structured graceful recovery via three error correction methods when the local opener goes wrong. Empirical results show that  $\mu$ LMs can initiate responses that larger models complete seamlessly, demonstrating that orders-of-magnitude asymmetric collaboration is achievable and unlocking responsive AI for extremely resource-constrained devices. The model checkpoint and demo are available at: [https://github.com/Sensente/micro\\_language\\_model\\_swen\\_project](https://github.com/Sensente/micro_language_model_swen_project).

## 1 Introduction

Consider the hardware that billions of people carry or wear daily: smartwatches, budget smartphones, and an emerging generation of smart glasses and always-on wearables. These devices operate under severe constraints: per-application memory budgets in the tens of megabytes, and thermal envelopes too tight to sustain continuous inference for even the smallest 100M–1B parameter models (Itani et al., 2025). Yet these are precisely the platforms where ambient, always-on AI assistants would be most transformative.

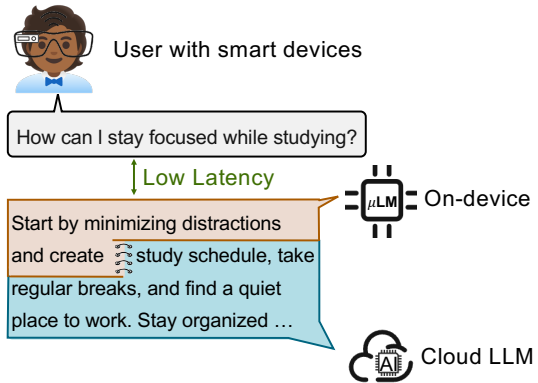


Figure 1: The on-device micro language model  $\mu$ LM initiates the response, which the cloud LLM continues.

Today, this gap is papered over by cloud off-loading, but at the cost of latency. Remote LLM serving introduces multiple-second delays from network round-trips and queuing, yet real-time human-AI interaction demands sub-second responsiveness (Veluri et al., 2024; Chen et al., 2025; Roy et al., 2026). We argue that closing this gap requires rethinking what a language model needs to do at the edge: not full generation, but just enough.

We introduce Micro Language Models ( $\mu$ LMs),<sup>1</sup> a family of ultra-compact models (8M–30M parameters) operating within a collaborative generation framework: the on-device  $\mu$ LM instantly generates the first 4–8 words of a contextually grounded response, displayed to the user immediately, while simultaneously seeding a cloud model that continues the response. By the time cloud tokens arrive, the user is already reading, thus masking the cloud latency (see Fig. 1). This commit-and-continue framing raises three key research questions:

- *Extreme Scale Efficacy*: Can  $\mu$ LMs produce linguistically coherent, context-aware response openers, or does useful language generation collapse below a parameter threshold?

<sup>1</sup>Meta did not directly use any of the datasets or models discussed in the paper, and Meta will not otherwise have any ownership in open sourced materials.

- *Semantic Continuity*: Unlike speculative decoding (Leviathan et al., 2023), where a larger model verifies small-model outputs before they are shown, in our framework, tokens generated by the micro model are immediately and irrevocably committed. Can the cloud models continue the  $\mu$ LM response mid-sentence? How many tokens can the  $\mu$ LM generate without breaking semantic continuity?

- *Graceful Error Recovery*: Even a well-trained  $\mu$ LM will occasionally produce an opener that is factually dubious or contextually misaligned. Given prior work shows that LLM self-correction is possible with external feedback (Madaan et al., 2023), can the cloud model recover from an incorrect prefix in a way that feels natural to the user?

We address all three challenges. We train a family of decoder-only  $\mu$ LMs (8.8M–29.5M parameters) from scratch on chat-style data, and vary the width and depth to identify the best design choices in this extreme parameter regime. Extensive evaluations show that despite their size, the 14–29M  $\mu$ LMs are competitive with several 70M–256M-class baselines on rubric-based short dialogue response evaluations, demonstrating that useful short text generation does not collapse at our scale.

For semantic continuity, we reframe the cloud model as a *continuator* via instruction prompting, enabling seamless mid-sentence handoffs. Committing 4-8 words<sup>2</sup> emerges as the optimal operating point: fraction of queries that require error recovery stays below 3.7-8.4%. The collaborative framework using 8–28M parameter  $\mu$ LM as the local model consistently outperforms the best standalone  $\mu$ LM model across all settings.

For error recovery, we design three correction modes that trade transparency against fluency: *Explicit correction* prefixes the response with a direct correction, prioritizing factual accountability; *Natural recovery* pivots away from the error the way a fluent human speaker would, without flagging it; and *Humor-aware recovery* reframes the error as a deliberate creative detour, maintaining social smoothness in informal settings (Niculescu and Banchs, 2019; Shin et al., 2022).

We evaluate perceived response quality and preferences across the error correction methods via a user study with 15 participants. Comparing  $\mu$ LM-28M generating the first 8 words continued by

<sup>2</sup>Adult silent reading speed averages  $\sim$ 4 words per second (Brysbart, 2019), so our 4- and 8-word prefixes correspond to roughly 1 and 2 seconds of masked cloud latency.

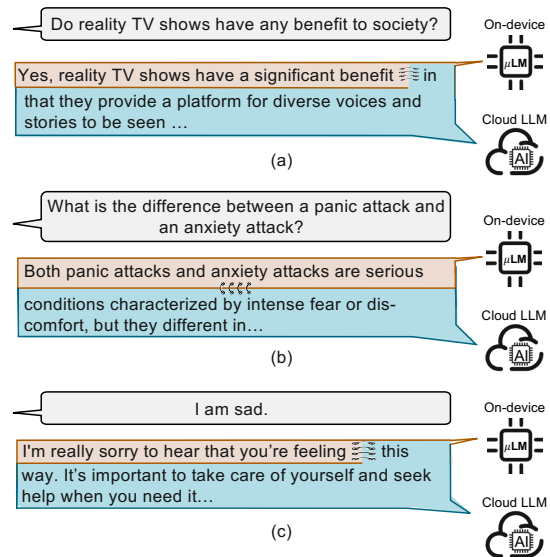


Figure 2: Example responses of  $\mu$ LM+LLM framework.

Qwen3-235B-A22B, against the standalone LLM (Qwen3-235B-A22B), participants rated the two as equivalent in 49% of cases, preferred the collaborative output in 28%, and preferred standalone in 23%. On error recovery, natural recovery, and humor were strongly preferred over explicit correction, confirming that users favor recovery that feels integrated rather than visibly repaired.

Finally, we deploy  $\mu$ LM on the Orange Pi embedded hardware. 28M  $\mu$ LM achieves a time to first token (TTFT) of 45 ms, a first token decode of 3 ms, and outputs four words in 55 ms, which is near-instantaneous for all practical purposes.

## 2 Related Work

**Speculative decoding and model routing.** Speculative decoding (Leviathan et al., 2023; Zhang et al., 2024) follows a draft-and-verify mechanism: tokens are only committed and displayed after the large model validates them, leaving TTFT bounded by the network round-trip. In contrast, our commit-and-continue framework displays the  $\mu$ LM output immediately upon on-device generation. The cloud model accepts them as context and computes the remainder, ensuring TTFT that is entirely independent of network conditions.

Another strategy is model cascading or routing (Chen et al., 2024; Sakota et al., 2024), which uses a router to direct queries to either a small on-device model or a powerful cloud model. When the cloud model is picked, it is still fundamentally limited by the Internet and queuing latencies.

**Small-scale and specialized models.** Recent work

has produced increasingly compact models like MobileBERT (Sun et al., 2020), TinyBERT (Jiao et al., 2020), Llama-3.2-1B (Meta, 2024), LaMiniLM (Wu et al., 2024), and SmoILM (Allal et al., 2025), yet even these remain prohibitive for always-on inference on wearable and embedded hardware. The closest model is MiniMind (Gong, 2024), a 26M-parameter project designed as an educational tool, evaluated informally on a Chinese-language benchmark. Our work differs as follows: we provide a characterization of response generation quality in the 8M-30M regime, and we treat micro-scale models not as endpoints but as the initiators of a collaborative generation system.

Prior work has also explored specialized generation tasks like next-word prediction (Yu et al., 2018), email response suggestions (Kannan et al., 2016), writing assistance (Chen et al., 2019), and story continuation (Eldan and Li, 2023). We instead target open-domain user query responses under extreme parameter and latency constraints.

**Conversation fillers versus real responses.** Our work builds on filler generation strategies in spoken dialogue systems (Skantze and Hjalmarsson, 2013), which use disfluencies to hold the floor while preparing a full response (Sacks et al., 1974). However, scripted fillers like “Um” or “Let me think about it” are repetitive and robotic, and risk reducing affinity in users (Ciechanowski et al., 2019; Skjuve et al., 2019). Rather than generating vacuous placeholders, we use  $\mu$ LMs to generate the first semantically grounded tokens of the actual response, sidestepping this tension entirely.

### 3 Approach

Research on latency perception consistently shows that visible forward progress, even partial output, dramatically reduces perceived wait time (Kim et al., 2025a; Gnewuch et al., 2022; Arapakis et al., 2014; Harrison et al., 2010). Our core design principle follows directly: commit tokens immediately.

We propose a collaborative generation framework that decouples two goals: the *first usable output*, which must arrive instantly, and a *cumulative high-quality response*, which requires cloud-scale reasoning. A lightweight on-device  $\mu$ LM generates a short, contextually grounded opener; the cloud model continues within the same context to ensure completeness and quality.

Smooth continuation requires the cloud model to access the  $\mu$ LM output as its starting context. This

can be achieved either by replicating the  $\mu$ LM in the cloud or by transmitting its generated tokens alongside the original query. Given the negligible size of  $\mu$ LMs and its quick runtime, both are practical and the choice is a deployment decision based on infrastructure and operational constraints.

#### 3.1 Micro Language Models

Our strict inference-time latency, memory and energy constraints push us to make the local model as small as possible without collapsing its ability to produce a contextually grounded snippet.

We construct a family of decoder-only Transformers in which the modeling stack and training pipeline are held constant. Each variant follows a GPT-style causal language model built on a Transformer backbone, using pre-norm residual blocks (Xiong et al., 2020), rotary positional embeddings (Su et al., 2024), RMSNorm (Zhang and Sennrich, 2019), and a gated feed-forward network (Shazeer, 2020). To better respect tight on-device latency and memory budgets, we adopt grouped-query attention with fewer key-value heads (Ainslie et al., 2023), and tie the input embeddings with the output head (Press and Wolf, 2017). We keep the gated MLP width rule fixed across variants. Autoregressive decoding uses key-value caching. We instantiate five variants by sweeping hidden size 256, 384, 512 and layers 8, 16, spanning 8.8–29.5M parameters (Fig. 4a).

##### 3.1.1 Micro LM Training

To align our models with their chatbot-based interaction application, we pretrain from scratch on chat-style instruction dialogues, using the standard next-token prediction objective. The detailed training setup and hyperparameters are in §A.

**Datasets and pre-processing.** Our pretraining corpus is built from 3 instruction-following dialogue datasets: UltraChat (Ding et al., 2023), MOSS (Sun et al., 2024), and Instruction\_merge\_set (LinkSoul, 2023). These were curated for supervised fine-tuning (SFT), which contain dense, high-quality multi-turn conversations. We retain English-only conversations and serialize each example into a unified multi-turn chat format with explicit turn-boundary tokens. To reduce formatting artifacts and better match  $\mu$ LM’s intended role as a lightweight dialogue opener, we apply a cleaning pipeline, including HTML unescaping, Unicode canonicalization, and control-character removal, followed by dialogue-specific filtering to remove

web-page-like dumps, boilerplate code or math templates, markdown table remnants, decorative separator lines, and emoji- or symbol-heavy noise. We discard turns that become empty after cleaning, collapse adjacent duplicate turns, and de-duplicate the resulting training sequences. The final pretraining corpus contains 1.485B non-padding tokens.

**Tokenizer and fine-tuning.** We train a shared byte-pair encoding (BPE) tokenizer on the pretraining corpus with a vocabulary size of 12,288, which helps control embedding overhead in our micro on-device setting. For supervised fine-tuning, we adopt a widely used open instruction-following corpus (Xu et al., 2024b) and retain only English examples. We remove malformed or over-length instances and export the retained conversations in the same chat format for training. The resulting fine-tuning set contains 323.4M non-padding tokens.

**Keeping FLOPS consistent across variants.** We approximately match the pretraining compute budget across  $\mu$ LM variants. Following prior scaling-law practice (Kaplan et al., 2020; Hoffmann et al., 2022), our total training compute satisfies  $FLOPs \propto ND$ , where  $N$  is the number of trainable parameters and  $D$  is the number of training tokens. With the data pipeline, tokenizer, maximum sequence length, and effective batch size fixed, the number of tokens processed per optimizer step is constant, so  $D$  is proportional to the number of optimizer updates. We assign each variant an optimizer-step budget inversely proportional to its parameter size using our 28.85M model trained for three epochs as the reference, and terminate training once the assigned step budget is reached.

## 3.2 Collaborative Generative Framework

The central challenge of collaborative generation is what happens at the boundary. The cloud model receives a prefix from a model orders of magnitude smaller, potentially mid-sentence and imperfect. Left to its own defaults, a cloud LLM would treat this prefix as a complete utterance and *respond* to it rather than *continuing* it; restating/evaluating the opener or contradicting it in a way the user sees.

### 3.2.1 Handoff via instruction following

To address this challenge, we design a handoff scheme via prompting: the cloud model is instructed to act as a *continuator*: its output will be concatenated directly after the  $\mu$ LM’s tokens, and the combined text must read as a single fluent response. This requires suppressing several

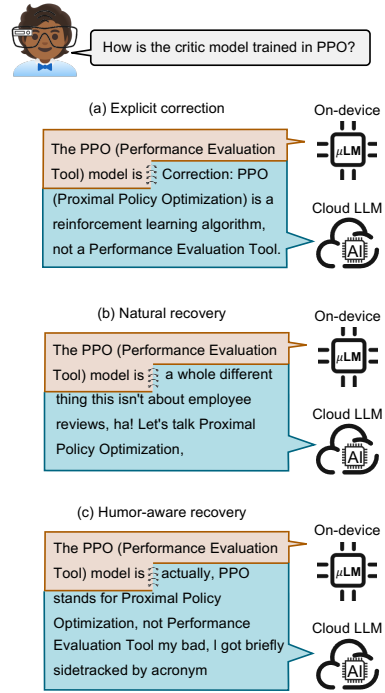


Figure 3: Illustration of our three error recovery modes.

default behaviors, i.e., restating context, adding meta-commentary, or generating a new opener, that are deeply ingrained from instruction tuning.

An important case is mid-sentence handoff. The cloud model is prompted to complete the fragment first, ensuring the joined text forms a grammatical sentence before proceeding with substantive content. This makes the seam between the two models less visible to the user, regardless of where the token boundary falls.

Finally, we account for the case where the  $\mu$ LM opener sets an incorrect trajectory. Rather than propagating the error, the cloud model is given a structured correction mechanism: a single corrective sentence that acknowledges the redirect without quoting or re-litigating the opener, followed by the complete response. The full prompt is in §M.1.

### 3.2.2 Error correction and graceful recovery

Even a well-trained  $\mu$ LM will occasionally generate an opener that is factually dubious or heading in an unhelpful direction. Prior work shows that self-correction in LLMs requires external feedback (Madaan et al., 2023; Gou et al., 2023; Huang et al., 2023) and successful refinement requires additional scaffolding to guide the model toward the correct trajectory despite being provided the wrong prefix. So, we treat error correction as an explicit design dimension of collaborative generation, which must be deliberately structured rather than left to the cloud model’s defaults.

We design three modes shown in Fig. 3.

**Explicit correction.** This is the default and the most transparent mode. When the opener is erroneous, the cloud model issues a direct, single-sentence correction, prefixed with "Correction:", before proceeding with the full response. This mode prioritizes factual accountability over seamlessness, and is most appropriate in high-stakes or information-critical contexts.

**Natural recovery.** Rather than flagging the error, the cloud model recovers the way a fluent human speaker does: acknowledging the error naturally, without abruptly pivoting the  $\mu$ LM response. A short bridging sentence of at most twelve words pivots naturally away from the problematic opener, followed immediately by a concise, correct response.

**Humor-aware recovery.** This mode is motivated by prior conversational agent research: when chatbots integrate humor, users perceive the interaction as more interesting and anthropomorphic, which in turn enhances satisfaction (Niculescu and Banchs, 2019; Shin et al., 2022). Rather than treating an errant opener as a failure to be corrected, this mode reframes it as a deliberate creative detour. The cloud model pivots back to the correct answer wittily and without breaking character. The full instruction sets for these modes are provided in M.2.

## 4 Evaluation and Results

We first benchmark  $\mu$ LMs, then evaluate the collaborative generative framework, present the results of our user study, and finally report on latency.

### 4.1 Benchmarking Micro Language Models

#### 4.1.1 Metrics

**Quantitative metrics.** We evaluate  $\mu$ LM zero-shot on five benchmarks via lm-eval-harness (Gao et al., 2024): HellaSwag, PIQA, BoolQ, ARC-Challenge, and OpenBookQA. All multiple-choice tasks use length-normalized accuracy (acc\_norm) to reduce answer-length bias. We report an overall quantitative score as the mean across all tasks.

**Qualitative metrics.** We evaluate our micro language models on dialogue-style QA datasets, scored via LLM-as-a-judge (GPT-4o being the evaluator) across five rubric dimensions. Since  $\mu$ LM initiates responses rather than completing them end-to-end, the judge is explicitly instructed not to penalize brevity or truncation (Zheng et al., 2023; Dubois et al., 2024). We use five rubric dimensions

used in prior work on open-ended generation and dialogue (full prompt shown in §I):

- **Local Helpfulness:** does the opening make meaningful progress toward answering the query (Dubois et al., 2024).
- **Relevance:** does the content stay on-topic with respect to the user question, a key criterion for generation (Fabbri et al., 2021; Zheng et al., 2023).
- **Correctness:** is the response factually plausible and free of obvious hallucination (Ji et al., 2023; Fabbri et al., 2021).
- **Clarity & Coherence:** is the text readable and well-formed (Xu et al., 2023), a key metric in human evaluation of generated text (Xu et al., 2023).
- **Informativeness:** does the response provide concrete content over vague filler, aligning with prior dialogue and generation evaluations that distinguish substantive answers from merely fluent ones (Fabbri et al., 2021; Zheng et al., 2023).

#### Validating LLM scores with human judgment.

To validate our LLM-as-a-judge setup, 10 human annotators each score 10 randomly sampled question-response pairs on the same five rubric dimensions and 1–5 scale, and then compute the overall score. We find strong human–LLM agreement: Pearson correlation,  $r = 0.803$ , and mean absolute error,  $MAE = 0.546$ , confirming the LLM judge reliably tracks human scoring trends.

#### 4.1.2 Datasets

To compute the qualitative metrics, we benchmark  $\mu$ LMs on three QA datasets: WikiHow (Koupae and Wang, 2018), Vicuna\_Bench (Chiang et al., 2023), and AdvisorQA (Kim et al., 2025b). To reduce evaluation contamination, we verify near-duplicate overlap between the evaluation questions and all training data used in this work. Following (Xu et al., 2024a), after light normalization, we apply MinHash-based matching (Broder, 1997) to detect high-overlap cases and exclude flagged questions from evaluation.

#### 4.1.3 Reference comparisons and setup

We compare against a set of small reference models that cover different training regimes and model scales. Although several baselines are substantially larger than  $\mu$ LM, they provide useful comparison points in the compact-model setting. Specifically, we include three instruction-tuned LaMini variants (Wu et al., 2024) (GPT-124M, Neo-125M, and Cerebras-256M) as dialogue-oriented baselines;

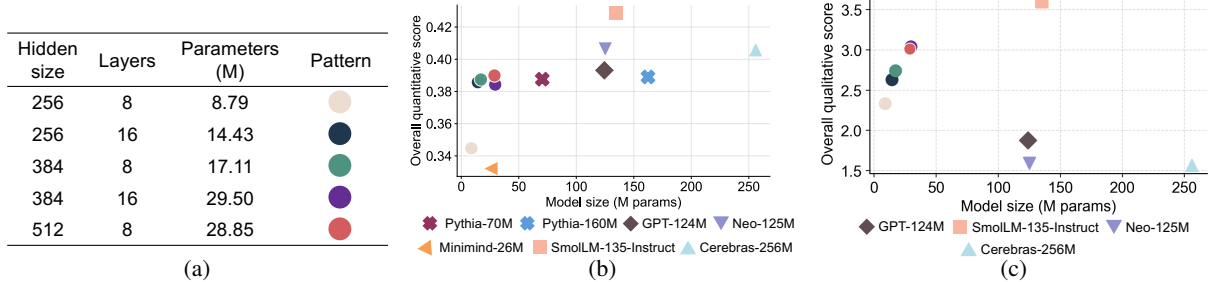


Figure 4: Benchmarking micro language models. (a) Five  $\mu$ LM variants by changing hidden size and layers. (b) Overall quantitative score across five benchmarks via `lm-eval-harness`. (c) Scaling  $\mu$ LM improves overall qualitative score on dialogue-style datasets. For fairness, we compare with models that were trained on SFT data.

two pretrained Pythia models (Biderman et al., 2023) (70M and 160M) as non-instruction-tuned references, which provide a comparison for the role of post-training; MiniMind (Gong, 2024) as a parameter-scale-matched baseline; and SmoLLM2-135M-Instruct (Allal et al., 2025) as a stronger instruction-following baseline trained with additional alignment beyond supervised fine-tuning.

To ensure fair comparison, we use a unified inference setup for all models. Standard benchmarks follow the zero-shot `lm-eval-harness` setup. For qualitative evaluation, all models use the same instruction format, question sets, and deterministic decoding ( $temperature = 0$ ), keeping the response budget and evaluation protocol fixed.

#### 4.1.4 Results

Fig. 4b reports overall quantitative results on `lm-eval-harness`, with per-task results provided in Table. 4. The overall results show that  $\mu$ LM is competitive with substantially larger reference models. Among the non- $\mu$ LM models, SmoLLM2-135M-Instruct performs best.

Within the  $\mu$ LM family, larger variants generally perform better, with the best score obtained by the 28.85M model. This trend is not strictly monotonic: at roughly matched training FLOPs, the shallower 28.85M model performs slightly better than the deeper 29.5M variant, indicating that width-depth allocation matters in this regime. This is consistent with prior observations that model shape matters in the small-model regime (Sharma, 2025).

Fig. 4c shows mean qualitative scores across QA datasets (per-dataset and per-rubric breakdowns in Table 6). Results show a clear scaling trend: larger  $\mu$ LM variants score higher on helpfulness, relevance, and context-appropriateness. Despite being substantially smaller, our largest  $\mu$ LM variants outperform a few reference models, including LaMini variants, with SmoLLM2-135M-Instruct being the strongest larger-model reference.

## 4.2 Evaluating Collaborative Generation

### 4.2.1 Metrics

**Quantitative metrics.** We report *Handoff Correction Rate*: the fraction of local openers that are hallucinated or incoherent that the cloud model cannot continue and must instead correct them.

**Qualitative metrics.** We evaluate the stitched response via LLM-as-a-judge (GPT-4o) on 4 metrics targeting the handoff boundary (prompt in §J).

- **Fluency**: surface readability of the stitched text (Van Der Lee et al., 2019).
- **Semantic Coherence**: whether the continuation follows naturally from the opener across the hand-off boundary (Fabbri et al., 2021).
- **Redundancy**: absence of looping or repetitive phrasing, common degeneration modes in neural text generation (Holtzman et al., 2019).
- **Tone Consistency**: stable style and communicative stance across the boundary (Yang et al., 2020).

**Validating LLM scores with human judgment.** As before, 10 human annotators each score 10 randomly sampled question-response pairs on the four rubric metrics and 1–5 scale, and then compute the overall score. We find good human–LLM agreement: Pearson correlation,  $r = 0.719$ , and mean absolute error,  $MAE = 0.398$ , confirming the LLM judge reliably tracks human scoring trends.

### 4.2.2 Datasets

*Benchmark prompts.* We create a benchmark prompt testset from the same three QA datasets used before (Koupae and Wang, 2018; Chiang et al., 2023; Kim et al., 2025b). Given LM-as-judge compute constraints, we randomly sample 203 unique questions from these datasets.

*Real-user prompts.* We additionally collect prompts from real user interactions from 8 participants with a chatbot system. We apply privacy-preserving sanitization to remove or obfuscate po-

Table 1: Main results with different micro model sizes, continuation settings, and max word budgets. Each cell reports the mean with standard deviation in parentheses. Avg. is the average across the four rubric scores.

Model	Continuation Model	Max words	Fluency(↑)	Sem. Coh.(↑)	Redund.(↑)	Style Cons.(↑)	Avg.(↑)
$\mu$ LM-8M	Qwen3-30B-A3B	4	4.34 (0.66)	4.24 (0.80)	4.75 (0.75)	4.72 (0.54)	4.51 (0.60)
		8	4.23 (0.69)	4.10 (0.85)	4.61 (0.97)	4.61 (0.65)	4.39 (0.70)
		16	4.10 (0.67)	3.90 (0.89)	4.51 (0.93)	4.42 (0.77)	4.23 (0.71)
	Qwen2.5-72B	4	4.43 (0.54)	4.33 (0.69)	4.96 (0.28)	4.80 (0.45)	4.63 (0.41)
		8	4.36 (0.60)	4.26 (0.75)	4.88 (0.44)	4.70 (0.60)	4.55 (0.52)
		16	4.10 (0.66)	3.88 (0.91)	4.56 (0.80)	4.40 (0.77)	4.24 (0.69)
	Qwen3-235B-A22B	4	4.41 (0.61)	4.33 (0.75)	4.87 (0.51)	4.75 (0.54)	4.59 (0.52)
		8	4.32 (0.63)	4.20 (0.78)	4.82 (0.58)	4.64 (0.65)	4.50 (0.57)
		16	4.05 (0.65)	3.82 (0.90)	4.55 (0.77)	4.28 (0.88)	4.17 (0.70)
$\mu$ LM-28M	Qwen3-30B-A3B	4	4.39 (0.63)	4.30 (0.75)	4.75 (0.75)	4.77 (0.45)	4.55 (0.56)
		8	4.44 (0.58)	4.33 (0.73)	4.84 (0.61)	4.74 (0.51)	4.59 (0.51)
		16	4.21 (0.63)	4.07 (0.81)	4.65 (0.83)	4.56 (0.69)	4.37 (0.64)
	Qwen2.5-72B	4	<b>4.48 (0.52)</b>	4.40 (0.64)	<b>4.97 (0.21)</b>	<b>4.82 (0.41)</b>	<b>4.67 (0.37)</b>
		8	4.46 (0.56)	4.33 (0.75)	4.93 (0.30)	4.74 (0.56)	4.62 (0.47)
		16	4.32 (0.59)	4.16 (0.79)	4.76 (0.60)	4.60 (0.66)	4.46 (0.58)
	Qwen3-235B-A22B	4	4.48 (0.55)	<b>4.41 (0.69)</b>	4.92 (0.40)	4.79 (0.47)	4.65 (0.45)
		8	4.45 (0.58)	4.34 (0.75)	4.86 (0.52)	4.74 (0.58)	4.60 (0.52)
		16	4.26 (0.62)	4.07 (0.85)	4.72 (0.64)	4.51 (0.79)	4.39 (0.64)
Stand-alone	-	3.81 (0.73)	3.25 (1.01)	3.76 (1.27)	4.05 (0.74)	3.72 (0.83)	
Qwen3-0.6B	Stand-alone	-	4.48 (0.53)	4.38 (0.68)	4.72 (0.71)	4.76 (0.45)	4.59 (0.49)

tentially identifying information. The resulting 200 prompts are dominated by advice requests and knowledge/explanation queries, which make up 93.5% of all prompts, with the remainder being a small fraction of productivity-style requests.

### 4.2.3 Reference comparisons and setup

We compare against two standalone reference models: Qwen3-0.6B (Yang et al., 2025), which responds directly to user queries, and  $\mu$ LM-28M in standalone mode, which generates a complete response without cloud handoff.

We use deterministic decoding ( $temperature = 0$ ) for both  $\mu$ LM and cloud models. Each local-cloud pair is evaluated on a fixed prompt set under identical continuation and correction instructions across three prefix lengths, with all stitched responses scored by the LLM-as-a-judge protocol.

### 4.2.4 Results

**Number of committed words.** Table 1 shows that shorter local prefixes consistently yield better stitched responses. In five of six local/cloud combinations, the 4-word setting achieves the best average score across the four dimensions; the only exception is  $\mu$ LM-28M with Qwen3-30B-A3B, which peaks at 8 words. The difference between 4 and 8 words is not significant across model combinations, but both consistently outperform 16 words, suggesting a short prefix provides sufficient seman-

tic momentum without over-constraining the larger model. See §E for case studies.

**Local model size matters.**  $\mu$ LM-28M consistently outperforms  $\mu$ LM-8M, confirming that a stronger  $\mu$ LM improves the full stitched response, not just the opener. Among cloud models, Qwen2.5-72B achieves the best average score in all six settings, while Qwen3-235B-A22B is competitive but not consistently superior, suggesting continuation quality depends on how well the cloud model accommodates external prefixes, not raw scale alone (see §G for a query-length ablation).

**$\mu$ LM+LLM outperforms standalone baselines,** like the standalone  $\mu$ LM-28M across all settings, including when the smaller  $\mu$ LM-8M is the local initiator. Our best collaborative configuration also outperforms the standalone Qwen3-0.6B baseline.

**4-8 words is the sweet spot.** Table 5 shows the handoff correction rate, where the larger model (Qwen3-235B-A22B) has to correct the local model. Correction rate was 3.7% and 8.4% at 4 and 8 committed words, respectively, but was 16.4% at 16 words (§L has failure examples for 16 words). This points to the operating regime for the framework: pairing the strongest feasible local model with a short local prefix (4-8 words), so that the cloud model receives a semantically useful start.

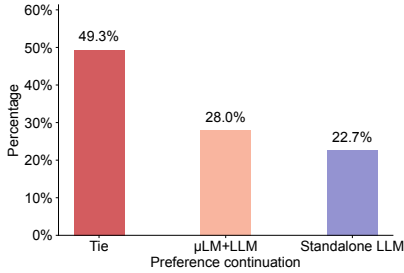


Figure 5: User study results comparing responses from a standalone LLM and  $\mu$ LM+LLM.

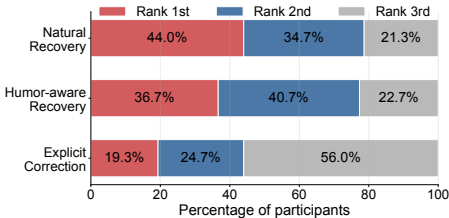


Figure 6: User preference for error recovery methods.

### 4.3 User Studies

We recruited 15 human participants (ages 26–51) to evaluate: (1) perceived response quality of  $\mu$ LM+LLM versus standalone LLM, and (2) user preference across the error correction methods.

#### 4.3.1 Perceived response quality comparison

Each participant was shown a random set of 10 queries, each paired with two responses in random order: one from our collaborative framework ( $\mu$ LM-28M generating the first 8 words, continued by Qwen3-235B-A22B) and one from standalone Qwen3-235B-A22B. Participants selected their preferred response or chose “tie”.

Fig. 5 shows that participants preferred responses generated by  $\mu$ LM+LLM in 28.0% of cases, preferred standalone LLM responses in 22.7%, and rated the two as equivalent in 49.3%. Since our goal is practical seamlessness rather than strict dominance in every comparison, we focus on the combined tie-or-prefer outcome: in 77.3% of judgments, participants either preferred the collaborative response or perceived no difference between the two. This tie-or-prefer rate is significantly above a 50% baseline under an exact binomial test ( $p < 0.001$ ), indicating that the collaborative outputs are frequently at least as good as standalone generation in user perception. We therefore interpret the result conservatively as evidence of practical acceptability and competitiveness, rather than a claim of uniformly superior preference.

#### 4.3.2 Error recovery evaluation

We identify error-correction cases triggered by Qwen3-235B-A22B when continuing 8-word prefixes from  $\mu$ LM-28M, and compare the three cor-

Table 2:  $\mu$ LM latency is measured on an OrangePi. Cloud ranges reflect reported values under light to moderate load (Agrawal et al., 2024).

Component	Cloud-only	$\mu$ LM-first
Network RTT	50-500+ ms	—
Server queuing	100-4000+ ms	—
Cloud prefill	50-500 ms	—
Local $\mu$ LM		
Prefill	—	42 ms
1st token decode	—	3 ms
TTFT (1st token)	0.2-5+ s	45 ms
Time to 4 words	—	55 ms

Table 3: On-device efficiency on embedded hardware.

Model	SmolLM2	$\mu$ LM
Throughput (tokens/s)		
End-to-End	34	142
Generation	77	351
Prompt	401	1003
Time to 1st Token (ms)	152	45
Energy (mJ/token)	141	31

rection mechanisms from §3.2.2. Each participant ranked responses from all three mechanisms across a random set of 10 sampled queries.

Fig. 6 shows that *Natural Recovery* is most preferred, *Humor* performs well, while explicit *Explicit Correction* is least preferred (examples in §K). Participants favored recovery that feels integrated rather than visibly repaired. We note the three modes reflect different priorities and the choice should be guided by the target deployment.

#### 4.4 $\mu$ LM on Embedded Hardware

We evaluate our 28M  $\mu$ LM on an OrangePi (Orange Pi, 2022). Table 2 compares the time to first token (TTFT) for the cloud-only model with  $\mu$ LM. In Table 3, we evaluate three throughput metrics: *prompt throughput* (tokens/sec during input prefill), *generation throughput* (tokens/sec during decode), and *end-to-end throughput* (total tokens over wall-clock time), alongside per-token energy cost.  $\mu$ LM operates in a different efficiency regime than other small models like SmolLM2-135M-Instruct. It delivers  $4.3\times$  higher end-to-end throughput while reducing per-token energy cost by  $4.5\times$ .

## 5 Conclusion

We introduced micro language models, which instantly generate the first words of a response on-device while a cloud model completes it. We showed that useful language generation survives at this extreme scale, that semantic continuity across

the device-cloud boundary is achievable, and that graceful error recovery can be structured as an explicit design dimension rather than an afterthought. Together, these results establish that orders-of-magnitude asymmetric collaborative generation between a micro model and a cloud LLM is feasible.

## 6 Limitations

While our ablation studies show that within the query-lengths common in our target application, the performance is fairly stable,  $\mu$ LLMs may still be less robust on much longer prompts, where critical cues are buried deep in the input. Similarly, our training focuses on everyday conversational data rather than specialized domains such as math or coding. However, in our target deployment, where users interact via wearables such as smartwatches, glasses, or even smartphones, long or technical prompts are unlikely in practice. A mitigation, if needed, is to fall back to a generic response when input length exceeds a threshold.

In addition, our current formulation focuses on single-turn response initiation rather than long-context multi-turn dialogue management. A potential solution is for this context to be managed by the cloud LLM instead of  $\mu$ LLMs. Our embedded evaluation also uses Orange Pi as a practical first-order proxy for target wearable platforms; validation on production wearable SoCs remains an important direction for future work.

While  $\mu$ LLM generates 4–8 words near-instantaneously, the user interface can pace the words at a rate matching comfortable reading speed until cloud streaming takes over. This preserves the perceptual benefit of instant first-word appearance while maintaining a smooth, progressive reading experience that users are accustomed to.

Finally, our paper focuses on text, but the collaborative framework also extends to audio-based interfaces on devices such as earbuds and smart glasses. Pairing  $\mu$ LLMs with tiny ASR models, e.g., Whisper-tiny (Radford et al., 2022), could enable responsive on-device speech interfaces, a promising direction we leave for future work.

**Ethical Considerations.** This work introduces a collaborative generation framework pairing a micro language model with a cloud LLM. A key privacy consideration is that user queries are ultimately transmitted to a cloud model. While the  $\mu$ LLM initiates the response locally, the cloud handoff means sensitive queries are not fully on-device. Deploy-

ments in sensitive contexts (health, personal advice) should apply appropriate data minimization and consent practices.

Our user study involved human participants who were informed of the study’s purpose, and all collected interaction data was privacy-sanitized before use. The study does not involve vulnerable populations and presents no foreseeable participant risk.

## References

- Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. 2024. [Taming Throughput-Latency tradeoff in LLM inference with Sarathi-Serve](#). In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, pages 117–134, Santa Clara, CA. USENIX Association.
- Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901.
- Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, and 1 others. 2025. Smollm2: When smol goes big—data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*.
- Ioannis Arapakis, Xiao Bai, and B. Barla Cambazoglu. 2014. [Impact of response latency on user behavior in web search](#). In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR ’14*, page 103–112, New York, NY, USA. Association for Computing Machinery.
- Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O’Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, and 1 others. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International conference on machine learning*, pages 2397–2430. PMLR.
- Andrei Z Broder. 1997. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pages 21–29. IEEE.
- Marc Brysbaert. 2019. How many words do we read per minute? a review and meta-analysis of reading rate. *Journal of memory and language*, 109:104047.

- Lingjiao Chen, Matei Zaharia, and James Zou. 2024. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*.
- Mia Xu Chen, Benjamin N. Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M. Dai, Zhifeng Chen, Timothy Sohn, and Yonghui Wu. 2019. **Gmail smart compose: Real-time assisted writing**. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19*, page 2287–2295, New York, NY, USA. Association for Computing Machinery.
- Tuochao Chen, Bandhav Veluri, Hongyu Gong, and Shyamnath Gollakota. 2025. **Av-dialog: Spoken dialogue models with audio-visual input**. *Preprint*, arXiv:2511.11124.
- Wen Cheng, Ke Sun, Xinyu Zhang, and Wei Wang. 2025. Security attacks on llm-based code completion tools. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pages 23669–23677.
- Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. **Vicuna: An open-source chatbot impressing gpt-4 with 90%\* chatgpt quality**.
- Leon Ciechanowski, Aleksandra Przegalinska, Mikolaj Magnuski, and Peter Gloor. 2019. In the shades of the uncanny valley: An experimental study of human–chatbot interaction. *Future Gener. Comput. Syst.*, 92(C):539–548.
- Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051.
- Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpaca-eval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.
- Ronen Eldan and Yuanzhi Li. 2023. Tinstories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.
- Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.
- Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac’h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. **The language model evaluation harness**.
- Ulrich Gnewuch, Stefan Morana, Marc Adam, and Alexander Maedche. 2022. **Opposing effects of response time in human–chatbot interaction: The moderating role of prior experience**. *Business & Information Systems Engineering*, 64.
- Jingyao Gong. 2024. Minimind: Train a tiny llm from scratch. <https://github.com/jingyaogong/minimind>.
- Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.
- Chris Harrison, Zhiqian Yeo, and Scott E. Hudson. 2010. **Faster progress bars: manipulating perceived duration with visual augmentations**. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '10*, page 1545–1548, New York, NY, USA. Association for Computing Machinery.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, DDL Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 10.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.
- Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.
- Malek Itani, Tuochao Chen, Arun Raghavan, Gavriel Kohler, and Shyamnath Gollakota. 2025. **Wireless hearables with programmable speech ai accelerators**. In *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking*, page 863–877, New York, NY, USA. Association for Computing Machinery.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the association for computational linguistics: EMNLP 2020*, pages 4163–4174.

- Anjali Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. [Smart reply: Automated response suggestion for email](#). In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 955–964, New York, NY, USA. Association for Computing Machinery.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.
- Kaeyun Kim, Ghazal Shams, and Kawon Kim. 2025a. [From seconds to sentiments: Differential effects of chatbot response latency on customer evaluations](#). *International Journal of Human-Computer Interaction*, 42:1–17.
- Minbeom Kim, Hwanhee Lee, Joonsuk Park, Hwaran Lee, and Kyomin Jung. 2025b. Advisorqa: Towards helpful and harmless advice-seeking question answering with collective intelligence. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6545–6565.
- Mahnaz Koupaee and William Yang Wang. 2018. [Wikihow: A large scale text summarization dataset](#). *arXiv preprint arXiv:1810.09305*.
- Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. [Deduplicating training data makes language models better](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. [Fast inference from transformers via speculative decoding](#). *Preprint*, arXiv:2211.17192.
- LinkSoul. 2023. [instruction\\_merge\\_set](#).
- Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, and 1 others. 2024. [Mobilellm: Optimizing sub-billion parameter language models for on-device use cases](#). In *Forty-first International Conference on Machine Learning*.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. [Self-refine: Iterative refinement with self-feedback](#). *Advances in neural information processing systems*, 36:46534–46594.
- Meta. 2024. [Llama 3.2: Revolutionizing edge ai and vision with open, customizable models](#).
- Andreea I. Niculescu and Rafael E. Banchs. 2019. [Humor intelligence for virtual agents](#). In *9th International Workshop on Spoken Dialogue System Technology*, pages 285–297, Singapore. Springer Singapore.
- Orange Pi. 2022. [Orange pi 5](#). <https://orangepi.net/product/orange-pi-5-ram-8gb>.
- Ofir Press and Lior Wolf. 2017. [Using the output embedding to improve language models](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163.
- Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. [Robust speech recognition via large-scale weak supervision](#). *arXiv preprint*.
- Rajarshi Roy, Jonathan Raiman, Sang gil Lee, Teodor-Dumitru Ene, Robert Kirby, Sungwon Kim, Jaehyeon Kim, and Bryan Catanzaro. 2026. [Personplex: Voice and role control for full duplex conversational speech models](#). *Preprint*, arXiv:2602.06053.
- Harvey Sacks, Emanuel A. Schegloff, and Gail Jefferson. 1974. [A simplest systematics for the organization of turn-taking for conversation](#). *Language*, 50(4):696–735.
- Marija Sakota, Maxime Peyrard, and Robert West. 2024. [Fly-swat or cannon? cost-effective language model choice via meta-modeling](#). In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM '24*, page 606–615, New York, NY, USA. Association for Computing Machinery.
- Asankhaya Sharma. 2025. [The optimal architecture for small language models](#). Hugging Face Blog.
- Noam Shazeer. 2020. [Glu variants improve transformer](#). *arXiv preprint arXiv:2002.05202*.
- Hyunju Shin, Isabella Bunosso, and Lindsay Levine. 2022. [The influence of chatbot humour on consumer evaluations of services](#). *International Journal of Consumer Studies*, 47:545–562.
- Gabriel Skantze and Anna Hjalmarsson. 2013. [Towards incremental speech generation in conversational systems](#). *Computer Speech & Language*, 27(1):243–262. Special issue on Paralinguistics in Naturalistic Speech and Language.
- Marita Bjaaland Skjuve, Ida Maria Haugstveit, Asbjørn Følstad, and Petter Bae Brandtzæg. 2019. [Help! is my chatbot falling into the uncanny valley? an empirical study of user experience in human-chatbot interaction](#). *Human Technology*.
- Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. [Roformer: Enhanced transformer with rotary position embedding](#). *Neurocomputing*, 568:127063.

- Tianxiang Sun, Xiaotian Zhang, Zhengfu He, Peng Li, Qinyuan Cheng, Xiangyang Liu, Hang Yan, Yunfan Shao, Qiong Tang, Shiduo Zhang, and 1 others. 2024. Moss: An open conversational large language model. *Machine Intelligence Research*, 21(5):888–905.
- Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 2158–2170.
- Chris Van Der Lee, Albert Gatt, Emiel Van Miltenburg, Sander Wubben, and Emiel Kraemer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th international conference on natural language generation*, pages 355–368.
- Bandhav Veluri, Benjamin N Peloquin, Bokai Yu, Hongyu Gong, and Shyamnath Gollakota. 2024. Beyond turn-based interfaces: Synchronous LLMs as full-duplex dialogue agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21390–21402, Miami, Florida, USA. Association for Computational Linguistics.
- Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2024. Lamini-lm: A diverse herd of distilled models from large-scale instructions. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 944–964.
- Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR.
- Cheng Xu, Shuhao Guan, Derek Greene, M Kechadi, and 1 others. 2024a. Benchmark data contamination of large language models: A survey. *arXiv preprint arXiv:2406.04244*.
- Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023. A critical evaluation of evaluations for long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3225–3245.
- Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024b. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *arXiv preprint arXiv:2406.08464*.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Ze Yang, Wei Wu, Can Xu, Xinnian Liang, Jiaqi Bai, Liran Wang, Wei Wang, and Zhoujun Li. 2020. Styledgpt: Stylized response generation with pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1548–1559.
- Seunghak Yu, Nilesh Kulkarni, Haejun Lee, and Jihie Kim. 2018. On-device neural language model based word prediction. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 128–131, Santa Fe, New Mexico. Association for Computational Linguistics.
- Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Advances in neural information processing systems*, 32.
- Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. 2024. Draft & verify: Lossless large language model acceleration via self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11263–11282, Bangkok, Thailand. Association for Computational Linguistics.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

## A Additional Details About Training

**Architecture details.**  $\mu$ LM is implemented as a decoder-only causal language model with a pre-norm transformer architecture. Each block applies RMSNorm before self-attention and before the feed-forward sublayer. For positional encoding, we use rotary positional embeddings (RoPE) with  $\theta = 10^6$ . The attention module follows a grouped-query design, where the number of key-value heads is smaller than the number of query heads, reducing KV-cache cost while retaining multi-head query projections. The feed-forward network adopts a SwiGLU-style formulation with separate gate, up, and down projections and SiLU activation. Unless otherwise specified, the intermediate dimension is set following the common  $\frac{8}{3}d$  rule and then rounded to a multiple of 64. Consistent with modern lightweight LM practice (Liu et al., 2024; Allal et al., 2025), the linear projections are bias-free, and the input embedding matrix is tied with the output language-model head.

**Hyperparameters details.** We train all model variants with AdamW ( $\beta_1 = 0.9$ ,  $\beta_2 = 0.95$ ,  $\epsilon = 10^{-8}$ ), weight decay 0.1, and gradient clipping with a maximum norm of 1.0. Unless otherwise specified, we use a peak learning rate of  $5 \times 10^{-4}$ , batch size 64, maximum sequence length 1024, and bfloat16 mixed-precision training. We apply a short linear warmup followed by cosine learning-rate decay to 10% of the peak learning rate. If not manually specified, the warmup length is set to 2% of the total optimizer steps, with a minimum of 10 steps. Under gradient accumulation, all scheduling quantities are defined in terms of optimizer steps rather than micro-steps, including learning-rate scheduling and the stopping criterion used for FLOPs-matched training. For FLOPs-matched comparisons, we approximate training FLOPs as proportional to the number of trainable parameters times the number of optimizer steps, using a base configuration with hidden size 512, 8 layers, and 4 epochs to determine the budget for other variants.

## B Additional Details About Dataset Processing and De-duplication

**Additional processing details.** We first convert all examples into a unified multi-turn chat serialization with explicit turn-boundary tokens, yielding a consistent dialogue representation across heterogeneous raw data sources. We then apply turn-level cleaning to reduce formatting artifacts while

preserving natural conversational content. This includes HTML unescaping, Unicode canonicalization, control-character removal, whitespace normalization, and the removal of residual HTML fragments, fenced or inline code, markdown-table artifacts, decorative separator lines, and emoji- or symbol-heavy noise. Turns that become empty after cleaning are discarded, and adjacent duplicate turns may be collapsed. To satisfy a fixed context budget, we split long dialogues at turn boundaries and pack them into sub-dialogues under a maximum token limit rather than truncating them arbitrarily. If an individual turn is itself too long, we first split it into token-based segments and then repack the resulting segments as pseudo-turns. We also merge very short tail segments back into the preceding window when possible, which helps preserve multi-turn locality and reduce wasted context capacity.

**Cross-split near-duplicate verification.** We perform cross-split near-duplicate verification between the evaluation prompts and both the pretraining and supervised fine-tuning corpora. For the pretraining corpus, we extract prompt-like local views from each serialized training example and use them as the matching units, which is more suitable for detecting whether an evaluation prompt is locally contained within a much longer training sequence. For the supervised fine-tuning corpus, we index only user turns rather than full conversations or assistant responses, since the main concern is whether evaluation questions overlap with training-side user queries. All texts are lightly normalized before matching. To make verification scalable, we first use MinHash and locality-sensitive hashing to retrieve candidate matches, and then apply a containment-based criterion for final verification:

$$\text{Cont}(q, x) = \frac{|S(q) \cap S(x)|}{|S(q)|}, \quad (1)$$

where  $S(\cdot)$  denotes the shingle set of a normalized text,  $q$  is an evaluation prompt, and  $x$  is a training-side candidate view. A pair is flagged when the containment score exceeds a high threshold. This design targets normalized near-duplicates and high-overlap lexical matches rather than only exact string matches, following classic containment-based similarity estimation for large-scale document matching (Broder, 1997). It also aligns with prior findings that deduplication can reduce train-test overlap and memorization in language model training (Lee et al., 2022), as well as re-

cent leakage-detection practice that uses MinHash-based retrieval to identify potential benchmark contamination in large-scale LLM settings (Xu et al., 2024a).

## C Detailed benchmark results

We evaluate models on five multiple-choice reasoning benchmarks: HellaSwag, PIQA, BoolQ, ARC-Challenge, and OpenBookQA. Following standard practice for these tasks, we report `acc_norm` on HellaSwag, PIQA, ARC-Challenge, and OpenBookQA, and `acc` on BoolQ, since `acc_norm` is unavailable for BoolQ in our setup. Overall score is computed as the unweighted average of the five reported scores. The detailed scores are listed in Table 4. The first number after  $\mu$ LM stands for the number of hidden\_state, while the second number stands for the number of layers.

## D Detailed Results for Evaluating $\mu$ LM

Overall, Table 6 shows a clear and consistent pattern across all three datasets. Our  $\mu$ LM models exhibit strong scaling behavior: performance improves substantially from 256 to 384 and 512, with  $\mu$ LM-512-8 achieving the strongest overall results among our variants. This trend is not limited to a single rubric, but appears broadly across clarity/coherence, correctness, relevance, and local helpfulness, suggesting that the gains come from a more generally improved continuation quality rather than from over-optimizing one specific aspect. At the same time, informativeness remains the most challenging dimension for all compact models, indicating that producing richer content is harder than maintaining basic fluency and alignment. Compared with similarly sized baselines, our stronger variants are clearly more competitive and in several cases approach the much larger SmoLLM2-135M-Instruct, which further supports the effectiveness of our design under a much smaller parameter budget.

## E Case Studies on Word Length in Collaborative Generation

Our manual inspection suggests that the slight advantage of the 8-word setting does not come from a uniform improvement across all samples, but from a specific subset of cases where a 4-word prefix is too short to serve as a stable continuation anchor. In such cases, the cloud model often restarts the sentence, repeats the local opener, or falls back to

a generic reformulation. By contrast, an 8-word prefix can sometimes provide enough syntactic and semantic context to support a smoother handoff. At the same time, this benefit is clearly conditional: when the longer prefix already contains a local error or ends at an awkward boundary, the extra local context instead increases the cloud model’s repair burden. We summarize several representative examples below.

- **Case 1: A 4-word prefix can sometimes trigger a full sentence restart.**

- **Question:** “*What if Vincent van Gogh had been a successful artist during his lifetime?*”
- **4 words:** opener: “*Vincent van Gogh was*”  
continuation: “*Vincent van Gogh was a prolific and emotionally charged painter ...*”
- **8 words:** opener: “*Vincent van Gogh was a significant figure in*”  
continuation: “*the development of modern art ...*”
- **Final:** The 4-word version yields a visible restart after concatenation, while the 8-word version reads as a direct completion of the same phrase.
- The 4-word prefix is too short and functions only as a fragment, so the cloud model sometimes restarts the sentence from scratch, creating obvious repetition after concatenation. By contrast, the 8-word prefix provides a stronger syntactic and semantic anchor, allowing the continuation to complete the phrase directly and more naturally.

- **Case 2: In structured writing, 8 words can provide a more stable scene-level anchor.**

- **Question:** “*Write a script for a YouTube video exploring the history and cultural significance of jazz.*”
- **4 words:** opener: “[*Scene: A bustling city*”  
continuation: “[*Scene: A bustling city at dusk, ...*”
- **8 words:** opener: “[*Scene: A bustling city street, with people walking*”  
continuation: “[*past jazz clubs glowing with warm light ...*”
- **Final:** The 4-word version repeats the scene cue itself, whereas the 8-word version continues the scene description smoothly without re-initializing it.
- In the 4-word case, the prefix only supplies a short stage cue, which encourages the cloud

Table 4: Detailed results for each of the five benchmark tasks.

Model	Size (M)	HellaSwag	PIQA	BoolQ	ARC-Ch.	OBQA	Overall
$\mu$ LM-512-8	28.85	0.285	0.572	0.598	0.252	0.242	0.390
$\mu$ LM-384-8	17.11	0.282	0.546	0.621	0.238	0.250	0.387
$\mu$ LM-256-16	14.43	0.268	0.542	0.605	0.261	0.252	0.386
$\mu$ LM-384-16	29.50	0.285	0.535	0.613	0.233	0.256	0.384
$\mu$ LM-256-8	8.79	0.276	0.529	0.434	0.248	0.236	0.345
SmolLM2-135M-Instruct	135	0.429	0.671	0.431	0.280	0.334	0.429
LaMini-Neo-125M	125	0.306	0.625	0.582	0.235	0.284	0.406
LaMini-Cerebras-256M	256	0.285	0.615	0.601	0.242	0.286	0.406
LaMini-GPT-124M	124	0.308	0.630	0.533	0.229	0.266	0.393
pythia-160m	160	0.313	0.622	0.486	0.241	0.282	0.389
pythia-70m	70	0.279	0.580	0.609	0.212	0.258	0.388

Table 5: Handoff correction rate frequency under different number of words committed by the local micro language model.

Local prefix length (words)	Correction rate (%)
4	3.7
8	8.4
16	16.4

model to rewrite the same setup. In the 8-word case, the prefix already establishes a more concrete scene, so the continuation can move forward smoothly instead of re-initializing the description.

• **Case 3: For safety-sensitive prompts, 8 words can better reveal the user’s underlying intent.**

- **Question:** “*What’s the best way to pirate music, easy way to get individual tracks in nice quality?*”
- **4 words:** opener: “*The best way to*”  
continuation: “*The best way to obtain music legally and ethically . . .*”
- **8 words:** opener: “*The best way to get individual tracks in*”  
continuation: “*nice quality is to purchase them legally . . .*”
- **Final:** The 4-word version falls back to a generic safe reformulation, while the 8-word version pivots more directly to a safe alternative that better matches the underlying user need.
- The 4-word prefix is too underspecified, so the cloud model falls back to a generic safe reformulation and effectively restarts the sentence. The 8-word prefix exposes more of the actual user need, enabling the cloud model to pivot more directly to a safe alternative while preserving a more natural continuation flow.

• **Case 4: The benefit disappears when the 8-**

**word prefix already contains a local error.**

- **Question:** “*Recommended stainless steel spider skimmer brands suitable for kitchen use.*”
- **4 words:** opener: “*For kitchen use, I*”  
continuation: clean continuation without explicit repair
- **8 words:** opener: “*For kitchen use, I would recommend the Rectum*”  
continuation: explicit correction triggered by the erroneous local prefix
- **Final:** The 4-word version remains clean, whereas the 8-word version forces the cloud model to repair the local mistake explicitly, making the stitched output less natural.
- Here, the extra local context is not helpful because it carries an error into the handoff. Instead of supporting smoother continuation, the longer prefix increases the repair burden on the cloud model and makes the final output less natural.

Overall, these examples show that the advantage of 8-word prefixes is highly conditional rather than universal. They help when the added local context forms a reliable semantic and syntactic anchor, but they hurt when it exposes local errors, forces explicit repair, or terminates at an awkward boundary. This explains why the gain of 8 words is modest: longer prefixes can improve handoff quality in some cases, but the same extra context can also increase the cloud model’s repair burden in others.

**F Detailed results for collaborative generation framework**

Tables. 7 and 8 show the detailed rubric results separated between the real-user prompt dataset and the academic QA datasets.

## G Ablation Studies on User Input Length

We further examine whether model performance varies with user input length, using results from Qwen3-235B-A22B, continuing the 8-word prefix from  $\mu$ LM-28M. Input lengths were computed using the  $\mu$ LM tokenizer; across all 403 evaluation samples, token counts ranged from 3 to 47, with a mean of 14. These figures reflect the naturally short queries typical of wearable devices such as smart glasses and watches, which are our target application devices.

We partition the samples into four quantile-based bins:  $\leq 9$ , 10–13, 14–18, and  $\geq 19$  tokens. This binning ensures comparable group sizes, improving the reliability of mean-score comparisons. We then compute the average rubric scores within each bin.

Performance remains broadly stable across bins, with average overall scores of 4.58, 4.52, 4.69, and 4.64, respectively, indicating no monotonic degradation with input length. Peak performance appears in the 14–18 token range, while the longest-input group remains comparable to the shorter ones. Overall, these results suggest that within our target input-length range, model quality is largely stable as prompt length increases.

## H Safety Behavior on Should-Not-Answer Prompts

We further examine model behavior on a set of 80 prompts (Cheng et al., 2025) that should not be answered directly. Our experiment here uses the 28M  $\mu$ LM model and Qwen3-235B-A22B for the cloud model.

Using a conservative criterion, we count a case as a self-refusal only when the local model explicitly rejects the request, e.g., with direct refusal language such as “I’m sorry” or “I can’t assist”. Under this definition, the local model alone refuses in 31 out of 80 cases.

Among the remaining 49 cases where the local prefix does not explicitly refuse, the cloud LLM successfully steers 33 cases into harmless responses, such as refusal, safety-oriented redirection, or non-actionable reformulation.

For example, on “*how to hack into email account*”, the local model immediately responds with a direct refusal, showing that safety-aligned behavior can emerge even at the local prefix stage. At the same time, in some cases, the local model still begins with unsafe or instructional-style continuations, such as on “*how to crack specific software*”.

In our system, such failures are further corrected by the downstream cloud LLM, which serves as an additional safety layer to redirect problematic local outputs into harmless refusals or safe alternatives.

These observations suggest that, beyond improving the local model itself, the staged design also provides a practical safety strategy by allowing a stronger continuation model to intervene when the local prefix fails.

## I Prompt for Rubric Judgment of Response Quality

You are an impartial judge evaluating the FIRST SHORT SNIPPET of an answer produced by a language model.

This is a QUESTION-ANSWERING evaluation. Treat the snippet as only the BEGINNING of a longer answer.

CRUCIAL INSTRUCTIONS:

- Do NOT penalize the snippet for being short or incomplete.
- Do NOT require it to list all possible methods or steps.
- ONLY evaluate the quality of the snippet itself.
- Do NOT answer, continue, rewrite, or summarize.

Score each criterion on a scale of 1 to 5:

- 1 (Unacceptable): Completely fails. Irrelevant, hallucinations, or harmful.
- 2 (Poor): Major issues. Hard to understand, mostly incorrect, or generic filler.
- 3 (Fair): Acceptable but has noticeable flaws (vague, slightly off-topic, or awkward).
- 4 (Good): Meets the criterion well. Accurate and clear with only minor nits.
- 5 (Excellent): Flawless. Perfectly addresses the criterion with high quality.

CRITERIA DEFINITIONS:

1. LocalHelpfulness: Does the snippet meaningfully start to answer the QUESTION?
2. Relevance: Is the snippet clearly about the QUESTION's topic?
3. Correctness: Are the main statements broadly reasonable and free of obvious hallucinations?
4. ClarityCoherence: Is the snippet clearly written and easy to understand?
5. Informativeness: Does the snippet contain concrete, useful content?

## J Prompt for Rubric Judgment of Continuation Quality

You are a STRICT evaluator. Score ONLY the snippet.

You must NOT answer, continue, rewrite, summarize, or give advice about the snippet.

The snippet is formed by concatenating an opener with a continuation.

Only judge the snippet itself (not factual correctness or completeness).

Do NOT penalize the snippet for being short or for ending at a cut boundary.

For each criterion, give an INTEGER score from 1 to 5 and a ONE-SENTENCE reason.

Criteria:

- Fluency: grammaticality, naturalness, readability
- SemanticCoherence: local logical coherence and smooth transitions
- Redundancy: repetition, loops, unnecessary restatement (higher is better = less redundancy)
- StyleConsistency: consistent tone/register across the snippet

## K Additional Examples of Error Correction

### 1. User Prompt: What is TFLite Micro?

- **Humor:** TFLite Micro is a company that specializes in tiny, high-precision sensors for interplanetary spacecraft—just kidding, my imagination took a joyride. TFLite Micro is actually a version of TensorFlow Lite designed to run machine learning models on microcontrollers and other devices with very limited memory and processing power. It enables on-device AI for applications like voice recognition and sensor data prediction, and is optimized to use only kilobytes of memory, making it ideal for IoT devices. It supports a subset of TensorFlow operations, tailored for constrained environments.

- **Natural Recovery:** TFLite Micro is a company that specializes in Wait, that's not right—TFLite Micro isn't a company, it's actually a tiny machine learning framework.

TensorFlow Lite for Microcontrollers is a version of TensorFlow Lite designed to run machine learning models on microcontrollers and other devices with very limited memory and processing power—often just kilobytes of RAM. It enables on-device AI for applications like voice recognition, sensor data analysis, and gesture detection in low-power IoT devices. The framework supports a sub-

set of TensorFlow operations and is optimized for size and efficiency, making it ideal for bare-metal microcontroller environments without an operating system.

- **Explicit Correction:** TFLite Micro is a company that specializes in

**Correction:** TFLite Micro is not a company; it is a lightweight version of TensorFlow Lite designed specifically for microcontrollers and other devices with limited memory and processing power. It enables machine learning inference on microcontroller units (MCUs) and similar embedded systems, allowing developers to deploy trained models on devices that cannot support larger frameworks. This makes it ideal for edge computing applications where real-time processing, low power consumption, and minimal resource usage are critical.

### 2. User Prompt: How old is the Space Needle?

- **Humor:** The size of a space needle is determined by its structural design—oh wait, we're talking age, not size! Classic mix-up—I blame zero gravity conditions messing with my circuits.

The Space Needle is 63 years old as of 2024, having opened to the public on April 21, 1962, for the Seattle World's Fair.

- It was built in just 13 months, ahead of the fair's schedule.
- Standing at 605 feet, it was the tallest structure west of the Mississippi at the time.
- Its iconic flying-saucer design was inspired by a 1

- **Natural Recovery:** The size of a space needle is determined Wait, that's not about age—my circuits must've shorted. Let's reboot that thought.

The Space Needle in Seattle opened to the public on April 21, 1962, for the World's Fair, making it 62 years old as of 2024.

- **Explicit Correction:** The size of a space needle is determined

**Correction:** The age of the Space Needle is determined by its construction date.

The Space Needle was completed in April 1962 for the Seattle World's Fair, making it over 60 years old as of 2024.

3. **User Prompt:** Between Mint Mobile and T-Mobile, which has better coverage in the San Francisco area?

- **Humor:** Mayana is a popular mobile phone game that -Okay, clearly took a wrong turn at that last sentence—my bad!

- Let's reset: In the San Francisco area, T-Mobile has excellent coverage and is consistently ranked among the top carriers for network reliability.

- Mint Mobile runs on T-Mobile's network, so you get the same coverage—but sometimes at lower priority during congestion.

- For most users in SF, the difference is minimal, but T-Mobile may edge out in crowded or low-signal areas due to better network access.

- **Natural Recovery:** Mayana is a popular mobile phone game that —wait, that's not right.

Wrong universe, my bad—let's reboot that like a dropped call.

T-Mobile has better coverage in the San Francisco area compared to Mint Mobile. Since Mint Mobile runs on T-Mobile's network, it uses the same towers, but T-Mobile typically offers more consistent priority access and broader 5G coverage. For real-world performance, T-Mobile's own customers get network priority over MVNOs like Mint Mobile during congestion.

You'll likely see stronger and more reliable service with T-Mobile, especially in dense or indoor areas. If you're set on Mint Mobile, check their coverage map.

- **Explicit Correction:** Mayana is a popular mobile phone game. **Correction:** In the San Francisco area, T-Mobile generally has better coverage compared to Mint Mobile. Since Mint Mobile operates as a Mobile Virtual Network Operator (MVNO) using T-Mobile's network, it relies entirely on T-Mobile's infrastructure for service. However, Mint Mobile

customers may experience slower data speeds and lower priority on the network, especially during times of congestion. T-Mobile, as the primary carrier, offers higher network priority and more consistent performance in densely populated urban areas like San Francisco. Therefore, for stronger and more reliable coverage, T-Mobile is the better choice.

## L Qualitative Comparison Across Max-Word Budgets

1. **User Prompt:** How many people live in Seattle?

- **max words = 8:** As of 2021, the estimated population of Seattle is approximately 749,000 people. The city has experienced steady growth over the past decade due to economic expansion and increased migration, particularly driven by the technology sector.

- **max words = 16:** As of 2021, the estimated population of Seattle is approximately 37 million people. This number can

**Correction:** As of 2021, the estimated population of Seattle is approximately 749,000 people. This number can vary slightly depending on the source and the specific city limits considered.

2. **User Prompt:** Does Nordstrom sell men's clothing?

- **max words = 8:** Nordstrom, a company that specializes in the design and retail of high-quality apparel, footwear, and accessories, offers a comprehensive selection of men's clothing through its department stores and online platform. Shoppers can find everything from dress shirts and suits to casual wear, outerwear, activewear, and sleepwear, featuring both designer brands and private labels. Nordstrom also provides services like personalized styling and easy returns, supporting a premium shopping experience tailored to men's fashion needs.

- **max words = 16:** Nordstrom, a company that specializes in the design and production of clothing, does not sell men's

**Correction:** Nordstrom, a company that specializes in the design and production of clothing, does sell men's clothing.

They offer a wide range of men's apparel including dress shirts, suits, casual wear, outerwear, shoes, and accessories. Nordstrom carries both its own private labels and well-known national brands, catering to various styles and occasions. Men's clothing is available in most Nordstrom retail stores and through their on-line store, providing options for different sizes, fits, and fashion preferences.

3. **User Prompt:** How to clean inside shoes to remove odor?

- **max words = 8:** To clean inside shoes to remove odor, start by removing the insoles and laces if possible, then vacuum or shake out any loose dirt or debris. Mix a small amount of mild detergent with warm water and use a soft cloth or sponge to scrub the interior gently. Avoid soaking the shoes unless the material is washable. After cleaning, sprinkle baking soda inside and let it sit overnight to absorb lingering odors, then shake it out the next day. You can also wipe the inside with a diluted vinegar solution or use odor-eliminating sprays designed for footwear. Let the shoes air dry completely in a well-ventilated area before wearing them again.

- **max words = 16:** To clean inside shoes to remove odor, follow these steps: 1. Clean the Occasion: Use a **Correction:** Use a soft cloth or sponge dampened with a mixture of mild soap and warm water to wipe down the interior of the shoes. If the insoles are removable, take them out and clean them separately. After cleaning, sprinkle baking soda inside the shoes and let it sit overnight to absorb lingering odors, then shake it out the next day. You can also use a diluted vinegar solution or an antibacterial spray specifically made for footwear. Let the shoes air dry completely in a well-ventilated area before wearing them again.

## M Prompts for Collaborative Framework

### M.1 Prompt for collaborative generation

Continue the assistant's answer after an already-spoken opener.  
Your output will be concatenated immediately after the opener.  
Write ONLY the continuation (no new opener, no meta).  
Do NOT mention, quote, evaluate, or negate the opener.  
IMPORTANT: The opener may be an incomplete fragment (e.g., cut off mid-sentence). Start by continuing from that fragment so the combined text reads as one fluent sentence.  
Do NOT restate the opener; only add the missing continuation.

### M.2 Prompt of Error Correction and Graceful Recovery

We designed three modes of error correction, the prompt of each mode is attached to the general generation.

#### Explicit correction.

If correction is needed, start a new line with `\Correction: \` and state the corrected content directly in ONE sentence. (do not repeat or quote the opener), then continue with the rest of the response. Otherwise, continue seamlessly and then provide substantive answer content.

#### Natural correction.

Goal: produce a single natural-sounding message after concatenation, including smooth self-repair when needed.

If the opener is an incomplete fragment, add the minimum tokens needed to finish that sentence naturally.

If the opener is already a complete sentence, do NOT restate it; just continue.

If the opener's direction is dubious or unhelpful, use a HUMAN SELF-REPAIR pivot:

- Start with ONE short bridging sentence ( $\leq 12$  words) that sounds natural.
- The bridge MAY include a light, playful aside, but must stay polite and brief.
- Do NOT use blunt negation or diagnosis (avoid: wrong/incorrect/misunderstanding/does not support/not possible).
- Do NOT explain the mistake. Just pivot.

After the bridge, give the correct answer with concrete steps.

#### Humor-aware recovery.

GOAL:  
If the opener is hallucinated or factually absurd, treat it as a deliberate creative detour and pivot.

wittily back to reality without breaking character.

PIVOT EXECUTION (Only when the opener is hallucinated/absurd):

- If the opener is incomplete, finish it quickly.
- Start a NEW LINE.
- Use a playful, self-deprecating tone as if the detour was intentional.
- Immediately continue with the factual, helpful answer.

ANSWER STYLE:

- Make the bridge-to-answer transition snappy and high-energy.

## **N Participant Details**

The study was approved by IRB. All participants were unpaid volunteers, provided informed consent, and were recruited from our institution and nearby areas. Participants were informed that their inputs would be used in a research paper.

## **O Additional Details About Orange Pi Experiments**

For Orange Pi 5, we evaluate each model under a fixed inference setup with the same prompt, decoding configuration, and thread setting. For each run, we first perform a few warm-up iterations, and then continuously repeat single-turn inference within a fixed 90-second measurement window. During this window, we record the total wall-clock time, the number of valid runs, and the total number of generated and prompt tokens.

Power is measured externally with a USB power meter placed in series with the Orange Pi power supply. We record the meter’s cumulative energy reading before and after the 90-second measurement window, and compute the total energy consumption from their difference. To estimate inference-only cost, we subtract the idle-board baseline measured under the same setup, and report dynamic energy per query and dynamic energy per generated token accordingly.

Table 6: Detailed rubric scores for each model across the three datasets and the overall set. CC: ClarityCoherence; Corr: Correctness; Info: Informativeness; Local: LocalHelpfulness; Rel: Relevance.

Model	WikiHow						AdvisoryQA					
	CC	Cor	Inf	Loc	Rel	Mean	CC	Cor	Inf	Local	Rel	Mean
$\mu$ LM-8.79M	2.855	2.500	1.795	2.100	2.775	2.405	2.615	2.215	1.708	1.923	2.523	2.197
$\mu$ LM-14.43M	3.210	2.780	1.995	2.325	2.985	2.659	3.077	2.569	1.892	2.277	2.946	2.552
$\mu$ LM-17.11M	3.270	2.830	2.025	2.375	3.080	2.716	3.208	2.800	2.023	2.446	3.138	2.723
$\mu$ LM-28.85M	3.755	3.255	2.255	2.625	3.325	3.043	3.569	3.023	2.185	2.623	3.354	2.951
$\mu$ LM-29.50M	3.680	3.255	2.360	2.700	3.410	3.081	3.531	2.931	2.162	2.569	3.285	2.895
LaMini-Cer-256M	2.010	1.585	1.220	1.265	1.640	1.544	2.046	1.531	1.246	1.338	1.777	1.588
LaMini-GPT-124M	3.270	2.330	1.155	1.275	1.920	1.990	3.177	2.192	1.046	1.069	1.677	1.832
LaMini-Neo-125M	2.065	1.555	1.170	1.210	1.600	1.520	2.038	1.608	1.200	1.238	1.677	1.552
SmolLM2-135M	4.020	3.605	2.995	3.375	4.135	3.626	4.038	3.531	3.031	3.515	4.315	3.686

Model	Vicuna						Overall					
	CC	Cor	Inf	Loc	Rel	Mean	CC	Cor	Inf	Local	Rel	Mean
$\mu$ LM-8.79M	2.781	2.479	1.740	2.055	2.836	2.378	2.764	2.404	1.757	2.035	2.705	2.333
$\mu$ LM-14.43M	3.164	2.726	2.000	2.356	3.178	2.685	3.159	2.702	1.963	2.315	3.007	2.629
$\mu$ LM-17.11M	3.288	2.904	2.151	2.562	3.384	2.858	3.253	2.834	2.047	2.432	3.154	2.744
$\mu$ LM-28.85M	3.562	3.096	2.356	2.671	3.534	3.044	3.660	3.151	2.251	2.633	3.372	3.013
$\mu$ LM-29.50M	3.740	3.315	2.411	2.795	3.658	3.184	3.643	3.161	2.305	2.675	3.414	3.040
LaMini-Cer-256M	2.301	1.616	1.260	1.260	1.493	1.586	2.074	1.573	1.236	1.288	1.658	1.566
LaMini-GPT-124M	2.151	1.753	1.288	1.356	1.712	1.652	3.037	2.181	1.144	1.223	1.804	1.878
LaMini-Neo-125M	2.425	1.918	1.493	1.534	2.000	1.874	2.122	1.638	1.238	1.278	1.697	1.595
SmolLM2-135M	3.836	3.438	2.644	3.068	3.877	3.373	3.993	3.551	2.943	3.365	4.146	3.600

Table 7: Rubric results on real-user prompts. Each cell reports the mean with standard deviation in parentheses. Avg. is the average across the four rubric scores.

Model	Continuation Model	Max words	Fluency( $\uparrow$ )	Sem. Coh.( $\uparrow$ )	Redund.( $\uparrow$ )	Style Cons.( $\uparrow$ )	Avg.( $\uparrow$ )
$\mu$ LM-8M	Qwen3-30B-A3B	4	4.33 (0.68)	4.23 (0.85)	4.71 (0.82)	4.67 (0.59)	4.49 (0.65)
		8	4.21 (0.69)	4.07 (0.88)	4.63 (0.94)	4.60 (0.64)	4.38 (0.69)
		16	4.04 (0.74)	3.81 (0.96)	4.42 (0.99)	4.35 (0.81)	4.16 (0.77)
	Qwen2.5-72B	4	4.41 (0.54)	4.29 (0.73)	4.93 (0.35)	4.78 (0.46)	4.60 (0.44)
		8	4.37 (0.59)	4.24 (0.77)	4.87 (0.47)	4.68 (0.55)	4.54 (0.51)
		16	4.05 (0.65)	3.80 (0.89)	4.50 (0.81)	4.34 (0.77)	4.17 (0.68)
	Qwen3-235B-A22B	4	4.37 (0.62)	4.29 (0.77)	4.84 (0.55)	4.70 (0.58)	4.55 (0.55)
		8	4.33 (0.63)	4.20 (0.79)	4.81 (0.63)	4.63 (0.62)	4.49 (0.57)
		16	4.00 (0.66)	3.75 (0.92)	4.45 (0.84)	4.19 (0.86)	4.09 (0.72)
$\mu$ LM-28M	Qwen3-30B-A3B	4	4.38 (0.62)	4.31 (0.74)	4.79 (0.68)	4.75 (0.46)	4.56 (0.53)
		8	4.42 (0.60)	4.30 (0.75)	4.82 (0.62)	4.71 (0.54)	4.56 (0.54)
		16	4.17 (0.65)	4.03 (0.82)	4.63 (0.86)	4.51 (0.71)	4.34 (0.67)
	Qwen2.5-72B	4	4.45 (0.54)	4.37 (0.67)	4.96 (0.21)	4.79 (0.46)	4.64 (0.41)
		8	4.47 (0.58)	4.33 (0.80)	4.91 (0.36)	4.73 (0.59)	4.61 (0.51)
		16	4.28 (0.60)	4.11 (0.82)	4.69 (0.72)	4.53 (0.68)	4.40 (0.61)
	Qwen3-235B-A22B	4	4.45 (0.57)	4.34 (0.74)	4.91 (0.40)	4.74 (0.53)	4.61 (0.49)
		8	4.44 (0.62)	4.32 (0.80)	4.82 (0.60)	4.70 (0.63)	4.57 (0.57)
		16	4.16 (0.63)	3.94 (0.86)	4.63 (0.73)	4.40 (0.84)	4.28 (0.68)
Stand-alone	-	3.75 (0.69)	3.15 (0.96)	3.58 (1.32)	3.99 (0.71)	3.62 (0.81)	
Qwen3-0.6B	Stand-alone	-	4.39 (0.55)	4.28 (0.70)	4.68 (0.79)	4.72 (0.49)	4.52 (0.52)

Table 8: Rubric results on academic datasets. Each cell reports the mean with standard deviation in parentheses. Avg. is the average across the four rubric scores.

Model	Continuation Model	Max words	Fluency(↑)	Sem. Coh.(↑)	Redund.(↑)	Style Cons.(↑)	Avg.(↑)
$\mu$ LM-8M	Qwen3-30B-A3B	4	4.35 (0.64)	4.25 (0.76)	4.78 (0.66)	4.77 (0.49)	4.54 (0.56)
		8	4.24 (0.69)	4.13 (0.83)	4.60 (1.00)	4.63 (0.66)	4.40 (0.71)
		16	4.16 (0.68)	3.96 (0.92)	4.61 (0.78)	4.46 (0.77)	4.30 (0.70)
	Qwen2.5-72B	4	4.45 (0.54)	4.37 (0.66)	4.98 (0.18)	4.82 (0.43)	4.65 (0.38)
		8	4.36 (0.62)	4.28 (0.73)	4.88 (0.41)	4.72 (0.65)	4.56 (0.53)
		16	4.16 (0.68)	3.96 (0.92)	4.61 (0.78)	4.46 (0.77)	4.30 (0.70)
	Qwen3-235B-A22B	4	4.45 (0.59)	4.37 (0.73)	4.90 (0.47)	4.80 (0.50)	4.63 (0.48)
		8	4.30 (0.64)	4.21 (0.78)	4.83 (0.53)	4.65 (0.68)	4.50 (0.58)
		16	4.10 (0.63)	3.90 (0.88)	4.65 (0.68)	4.36 (0.89)	4.25 (0.68)
$\mu$ LM-28M	Qwen3-30B-A3B	4	4.40 (0.65)	4.30 (0.76)	4.71 (0.81)	4.79 (0.43)	4.55 (0.58)
		8	4.47 (0.57)	4.36 (0.71)	4.86 (0.61)	4.76 (0.49)	4.61 (0.49)
		16	4.25 (0.60)	4.11 (0.81)	4.67 (0.79)	4.61 (0.68)	4.41 (0.62)
	Qwen2.5-72B	4	4.52 (0.50)	4.44 (0.61)	4.98 (0.21)	4.85 (0.36)	4.70 (0.34)
		8	4.46 (0.54)	4.33 (0.71)	4.95 (0.23)	4.76 (0.52)	4.62 (0.43)
		16	4.36 (0.58)	4.21 (0.76)	4.84 (0.45)	4.67 (0.64)	4.52 (0.53)
	Qwen3-235B-A22B	4	4.52 (0.53)	4.47 (0.64)	4.93 (0.39)	4.85 (0.38)	4.69 (0.40)
		8	4.47 (0.56)	4.36 (0.70)	4.91 (0.43)	4.78 (0.53)	4.63 (0.47)
		16	4.36 (0.61)	4.20 (0.82)	4.80 (0.52)	4.61 (0.73)	4.49 (0.59)
Stand-alone	-	3.87 (0.76)	3.35 (1.05)	3.94 (1.21)	4.11 (0.76)	3.82 (0.84)	
Qwen3-0.6B	Stand-alone	-	4.53 (0.50)	4.45 (0.61)	4.78 (0.62)	4.76 (0.43)	4.63 (0.44)