An Empirical Study of On-Device Translation for Real-Time Live-Stream Chat on Mobile Devices

Jeiyoon Park[1,†], **Daehwan Lee**[1], **Changmin Yeo**[1], **Yongshin Han**[1], **Minseop Kim**[1],
[1]SOOP
k4ke@korea.ac.kr, {ldh, yeolife, winterfeb, usio}@sooplive.com
https://huggingface.co/datasets/naruto-soop/LiveChatBench

Abstract

Despite its efficiency, there has been little research on the practical aspects required for real-world deployment of on-device AI models, such as the device's CPU utilization and thermal conditions. In this paper, through extensive experiments, we investigate two key issues that must be addressed to deploy on-device models in real-world services: (i) the selection of on-device models and the resource consumption of each model, and (ii) the capability and potential of on-device models for domain adaptation. To this end, we focus on a task of translating live-stream chat messages and manually construct LIVECHATBENCH, a benchmark consisting of 1,000 Korean–English parallel sentence pairs. Experiments on five mobile devices provide a systematic empirical assessment of widely adopted on-device models, highlighting the importance of model selection and deployment constraints when adapting them to specialized tasks and serving a large and heterogeneous user base. We expect that our findings will offer practical insights into both the capabilities and limitations of on-device models for real-world AI service deployment.

1 Introduction

Large language models (LLMs) have driven major advances across a wide range of AI applications, while small language models (SLMs) have also shown significant promise by achieving competitive performance with substantially lower computational and memory demands (Xu et al., 2024; Bohdal et al., 2025; Liu et al., 2024; Allal et al., 2025; Pham et al., 2025; Hau et al., 2025; Zhao et al., 2025). However, although real-world deployment of on-device models requires accounting for heterogeneous device conditions and user environments, existing studies have paid limited attention to practical operational constraints, such as CPU resource consumption and fluctuations in device temperature. Though (Liu et al., 2024) and (Pham et al., 2025) address practical challenges, they focus only on model architecture or training.

In this paper, we examine the challenges that must be addressed to deploy on-device models in real-world services, focusing on selecting appropriate models, characterizing the resource usage of each, and assessing their capacity and potential for domain adaptation. Inspired by a culturally-aware evaluation benchmark (Kim et al., 2024), we manually construct LiveChatBench, a high-quality benchmark consisting of 1,000 Korean–English live-streaming chat translation pairs that include memes, slang expressions, and ungrammatical or ill-formed text patterns, along with annotations of the knowledge required for accurate translation.

We conduct two types of experiments using three SLMs with 270M, 0.6B, and 1B parameters (Team et al., 2025; Yang et al., 2025), three iOS devices, and two Android devices: (i) we measure mobile-device CPU utilization, temperature variations, time to first token, and runtime on LiveChatBench under CPU-only execution and under GPU-accelerated execution, and (ii) we evaluate the domain adaptation performance of on-device models on LiveChatBench, FLORES-200 (Team et al., 2022), and WMT++ (Deutsch et al., 2025).

Results demonstrate that, given the performance constraints of client devices and the need to preserve user convenience, deploying on-device models in real-world applications inevitably requires making a limited set of design trade-offs; nevertheless, through the lens of domain adaptation, the proposed approach achieves performance comparable to that of commercial models such as GPT-5.1 on the well-targeted task, highlighting both the effectiveness and the latent potential of on-device AI. We believe our findings offer insights into the unavoidable challenges faced by researchers designing models under strict user-side resource

[†]The first author is currently at Smilegate.

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 1–10
November 6-10, 2026 ©2026 Association for Computational Linguistics

Source (ko)	Target (en)	Background Knowledge
메접 빌드업 지린다	This build-up to quitting MapleStory is insane.	**메접**: '메접'은 온라인 게임 메이플스토리에서 접는다 (그만둔다)는 뜻으로 쓰이는 줄임말입니다. **빌드업**: '빌드 업(build up)'은 '쌓아 올리다', '점점 증가시키다'라는 뜻으로, 어떤 것을 단계적으로 만들어가는 과정을 의미합니다.
애가 **억타**인데	The kid is being forced to play StarCraft, though.	**억타**: '억지로 스타크래프트를 한다'라는 의미. '억마크'처럼 '억+(게임 이름)'으로 응용된다.
어제 **비챤**님 롤대회가 인기많던데..	Yesterday, VIichan's League of Legends tournament seemed really popular.	**비챤**: 인터넷 방송 플랫폼 "SOOP (숲)"의 스트리머 이름.

Table 1: Examples from the LiveChatBench dataset.

	BM25	LLM	BM25 + LLM
Micro Recall (↑)	0.4834	0.7031	**0.8107**

Table 2: **Micro-averaged recall results.** The methods compared are BM25, LLM-based entity extraction (using GPT-5.1), and a hybrid BM25+LLM approach.

constraints, including limited battery capacity (Liu et al., 2024; Malladi et al., 2012; Han et al., 2016).

2 Datasets

2.1 Data Collection

First, we collect approximately 30 million chat messages from a live-streaming platform[1,2]. Then, we filter out overly uninformative instances (e.g., *"ㅋㅋㅋ"*, meaning *"lol"*) and messages longer than 50 characters, since our analysis of the collected livestream chat data showed that 99.03% of messages were within this length. Synthetic parallel pairs are generated using the pipeline described in Section 2.2. We finally construct a dataset comprising approximately 1.5 million training and development examples in total. Note that this dataset contains a wide range of memes, slang expressions, and ungrammatical or ill-formed text patterns, which can substantially degrade an LLM's ability to interpret and understand the content (Sun et al., 2022, 2024; Wuraola et al., 2024).

2.2 LiveChatBench

Building on previous studies showing that synthetic data can improve translation performance (Kartik et al., 2024; de Gibert et al., 2025), we use an LLM to translate the dataset collected in Section 2.1. However, simply constructing a synthetic dataset is more likely to inject erroneous knowledge because chat data are difficult for even state-of-the-art LLMs to interpret. Since knowledge injection greatly facilitates the construction of synthetic data (Shen et al., 2025), we first manually build a dictionary of internet terminology and slang required for translation, consisting of 656 words, and then adopt a framework that retrieves and incorporates this dictionary during the data generation process.

Human Annotations. To evaluate data generation pipelines and trained on-device translators, we invite annotators and build LIVECHATBENCH, a high-quality benchmark of 1,000 chat instances, each paired with translation outputs and the necessary translation-relevant knowledge (Table 1).

Validation. As shown in Table 2, LiveChatBench offers the advantage of enabling us to verify whether the currently constructed pipeline is effectively injecting knowledge. In this paper, we employed BM25 and an LLM-based entity extractor for keyword-based knowledge injection.

3 Experimental Setup

All experiments were conducted using five smartphones: iPhone 11 Pro, iPhone 14 Pro Max, iPhone 16 Pro Max, Samsung Galaxy S24+, and Samsung Galaxy S25. We leveraged GPT-5.1 to construct the training dataset. An NVIDIA H100 NVL GPU was used to train the on-device models.

3.1 Mobile Device Specifications

Table 3 shows the specifications of the five mobile devices used in the experiment. For iOS deployments, we utilize Apple's Metal API for GPU acceleration through llama.cpp's Metal backend[3]. We did not run GPU experiments on the iPhone 11 Pro because the llama.cpp Metal backend requires SIMD-group operations that are available only on devices supporting Apple GPU Family 7 or later (as specified in Apple's Metal feature set tables), which the iPhone 11 Pro does not support.

3.2 Baselines

To investigate whether on-device AI can be applied to real-world applications, we select four models: MLKit (Google, 2019), Gemma-3-270M (Team et al., 2025), Qwen3-0.6B (Yang et al., 2025), and Gemma-3-1B (Team et al., 2025). We also choose two models to measure the gap between commer-

[1] https://www.sooplive.co.kr/

[2] All data are collected and used in accordance with the relevant usage permissions, and no personally identifiable information is included. The dataset consists of Korean chat data and is used exclusively for academic research.

[3] https://github.com/mybigday/llama.rn

	CPU	GPU	RAM
iPhone 11 Pro	2-core Apple Lightning 2.67 GHz, 4-core Apple Thunder 1.73 GHz	4-core Apple 3rd-generation design GPU architecture, 0,000 MHz	4 GB LPDDR4X SDRAM
iPhone 14 Pro Max	2-core Apple Everest 3.46 GHz, 4-core Apple Sawtooth 2.02 GHz	5-core Apple G14, 1,398 MHz	6 GB LPDDR5 SDRAM
iPhone 16 Pro Max	Dual-core Apple Everest (3rd generation) 4.04 GHz, Quad-core Apple Sawtooth (3rd generation) 2.40 GHz	6-core Apple G17, 1,470 MHz	8 GB LPDDR5X SDRAM
Galaxy S24+	1 × ARM Cortex-X4 3.21 GHz, 2 × ARM Cortex-A720 2.90 GHz, 3 × ARM Cortex-A720 2.60 GHz, 4 × ARM Cortex-A520 at 1.95 GHz	Samsung Xclipse 940 1.1 GHz	12 GB LPDDR5X SDRAM 8,533 MT/s
Galaxy S25	2 × Oryon-L (2nd Gen) 4.47 GHz, 6 × Oryon-M (2nd Gen) 3.53 GHz	Qualcomm Adreno 830 (1.2 GHz)	12 GB LPDDR5X SDRAM 9,600 MT/s

Table 3: **Specifications of the mobile devices used in our experiments, including three iOS devices and two Android devices.** On iOS devices, we used Apple's Metal framework for GPU-accelerated inference.

	Model Weights	KV Cache	Compute Buffer	Total
Gemma-3-270M	511 MB	192 MB	18 MB	721 MB
Qwen3-0.6B	1137 MB	112 MB	112 MB	1361 MB
Gemma-3-1B	1907 MB	26 MB	192 MB	2125 MB

Table 4: **Component-wise memory breakdown.** We report the total memory footprint of the Gemma-3-270M, QWEN3-0.6B, and Gemma-3-1B models, decomposed into model weights, KV cache, and compute buffers.

Parameter	Value
`n_ctx`	512
`n_threads`	2
`n_batch`	256
`n_uBatch`	192
`flash_attn_type`	`On`
`n_predict`	128
`temperature`	0.01
`top_p`	0.95

Table 5: **Model and completion parameters.** We employ a unified configuration for all on-device models.

cial models and on-device models: Google Translate API[4] and GPT-5.1 (OpenAI, 2025).

3.3 Implementation Details

Training Configuration. This section details the experimental setup and the hyperparameter settings used throughout this work. Table 8 summarizes the common training parameters that are uniformly applied across all on-device models and datasets.

Inference. To improve efficiency, we use a simple prompt for inference: *"Translate the following sentence into English. Only include the translated result, do not explain the result."*. Table 5 shows the model parameters and completion parameters used for inference on each mobile device.

[4] https://pypi.org/project/googletrans/

3.4 Evaluation Protocols

We evaluate translation performance on three Korean–English benchmarks: LiveChatBench (1,000 parallel sentence pairs; hereafter, parallel pairs), FLORES-200 (1,012 parallel pairs) (Team et al., 2022), and WMT24++ (998 parallel pairs) (Deutsch et al., 2025). We report BLEU (Papineni et al., 2002), ChrF++ (Popović, 2015), and use GPT-5.1 to assess six translation error types via the Focus Sentence Prompting format (Domhan and Zhu, 2025), following (Li et al., 2025) (Figure 7).

4 On-Device Performance Results

GPU acceleration is a key enabler for real-world on-device applications. We evaluate the thermal state and CPU utilization of three iOS devices and two Android devices on LiveChatBench. Since Apple does not expose numerical device temperatures to iOS developers, we report the thermal states of iOS devices using Apple's four-level classification. The *nominal* state corresponds to temperatures within normal limits, while *fair* denotes slightly elevated temperatures. The *serious* and *critical* states both reflect high-temperature conditions that significantly affect overall system performance and require the device to cool down immediately.

Figures 1(a)-(c) show that thermal conditions remain stable across devices when GPU accelera-

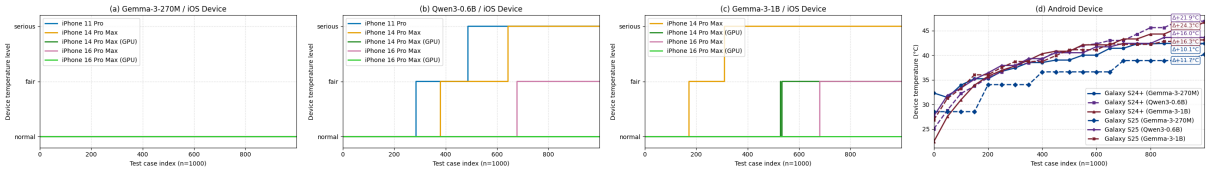


Figure 1: Mobile device temperature results on LiveChatBench across different on-device models.

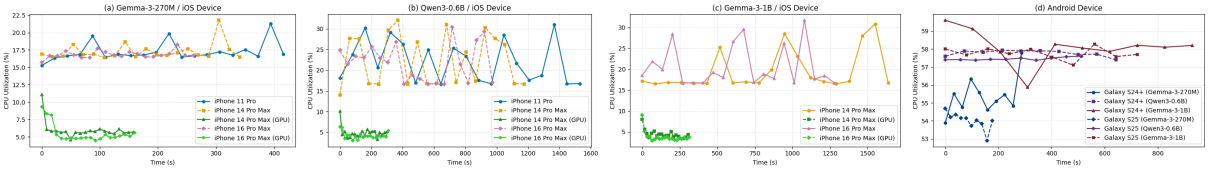


Figure 2: CPU utilization results on LiveChatBench across different on-device models.

Mobile Device	TTFT (ms)	Runtime (s)
Gemma-3-270M-IT†		
iPhone 11 Pro	140.9460	0.4161
iPhone 14 Pro Max	121.4954	0.3412
iPhone 14 Pro Max‡	74.7435	0.1582
iPhone 16 Pro Max	79.3681	0.2608
iPhone 16 Pro Max‡	63.3153	0.1598
Samsung Galaxy S24+	48.0430	0.2868
Samsung Galaxy S25	38.4525	0.1769
Qwen3-0.6B†		
iPhone 11 Pro	811.4484	1.5294
iPhone 14 Pro Max	637.4881	1.1753
iPhone 14 Pro Max‡	158.7082	0.3022
iPhone 16 Pro Max	460.8820	0.9730
iPhone 16 Pro Max‡	149.2322	0.3104
Samsung Galaxy S24+	186.3310	0.6413
Samsung Galaxy S25	189.9350	0.5126
Gemma-3-1B-IT†		
iPhone 14 Pro Max	784.0577	1.6371
iPhone 14 Pro Max‡	159.3810	0.3046
iPhone 16 Pro Max	536.1123	1.2821
iPhone 16 Pro Max‡	151.2176	0.3126
Samsung Galaxy S24+	230.3490	0.9283
Samsung Galaxy S25	240.3030	0.7219

Table 6: **Evaluation of three on-device models on LiveChatBench across five mobile devices.** † denotes models trained on the dataset constructed in Section 2. ‡ indicates results measured in an environment where the mobile device's GPU was used.

tion is enabled. As shown in Figures 2(a)-(c), CPU utilization, an important metric for real-world applications, remains around 5% with minimal fluctuation under GPU acceleration. These results highlight the importance of considering users' device environments when deploying on-device models as a service, as some devices do not support GPU acceleration (e.g., the iPhone 11 Pro in Table 3). Although the Galaxy S24+ and Galaxy S25 achieve fast average runtimes (Table 6), their temperatures rise above 45°C and CPU utilization approaches 60% under CPU-only execution, as shown in Figure 1(d) and Figure 2(d), respectively. These thermal and computational demands may hinder the practical deployment of on-device models in real-world services without GPU acceleration.

On-device model selection is significantly constrained in mobile environments. Table 6 shows the average time to first token (TTFT) and runtime on LiveChatBench for three on-device models across three iOS devices and two Android devices. We observe that, in the GPU environments of the devices, trained Gemma-3-270M exhibits an average runtime that is approximately 1.9178× faster and a average TTFT that is approximately 2.2503× faster than those of Qwen3-0.6B and Gemma-3-1B, demonstrating markedly better suitability for rapidly translating users' chat messages.

Considering that a fully charged iPhone 16 Pro Max stores 65,376 J of energy, Gemma-3-270M and Gemma-3-1B, which consume 0.027 J and 0.1 J per token respectively, can generate up to 2,421,333 and 653,760 tokens, corresponding to a factor of approximately 3.703× (Malladi et al., 2012; Han et al., 2016). Moreover, the iPhone 11 Pro cannot run a 1B-parameter model due to its limited memory capacity (Table 4). Therefore, to accommodate users with diverse devices, task clarification, the specifications of devices, and the selection of model size must be carefully considered.

5 Translation Results

On-device models demonstrate excellent domain adaptation performance. Table 7 demonstrates the translation performance of each model on three Korean–English translation benchmarks. We ob-

Method	LiveChatBench			FLORES-200			WMT24++		
	BLEU (↑)	ChrF++ (↑)	FSP (↑)	BLEU (↑)	ChrF++ (↑)	FSP (↑)	BLEU (↑)	ChrF++ (↑)	FSP (↑)
MLKit	0.0489	17.4392	28.2600	0.2556	44.9525	52.5267	0.1737	39.5039	50.3096
Google Translate API	0.1678	34.2974	59.7250	**0.4449**	**60.5056**	94.1294	**0.3333**	**53.5668**	91.7806
GPT-5.1	0.2679	45.2609	**70.2210**	0.4113	59.0201	**96.8607**	0.2998	50.8218	**96.2745**
Gemma-3-270M-IT	0.0097	5.3872	15.1300	0.0740	20.9606	23.8636	0.0553	17.4603	23.1293
Qwen3-0.6B	0.0017	8.6502	5.9980	0.0261	21.9348	4.8715	0.0221	18.2811	5.7615
Gemma-3-1B-IT	0.0430	17.5645	35.2680	0.2301	42.4403	65.5188	0.1435	33.4784	59.9639
Gemma-3-270M-IT†	0.2485	43.9363	62.9280	0.1328	32.0134	31.7738	0.1102	30.5162	40.7335
Qwen3-0.6B†	0.2689	45.7545	65.9700	0.1794	37.7578	47.2826	0.1402	34.4146	50.4719
Gemma-3-1B-IT†	**0.2978**	**48.4404**	67.8740	0.1978	39.3742	53.2401	0.1556	36.4609	54.7896

Table 7: **Comparison of model performance across three different translation datasets.** Higher values indicate better translation quality. † denotes models trained on the dataset constructed in Section 2.

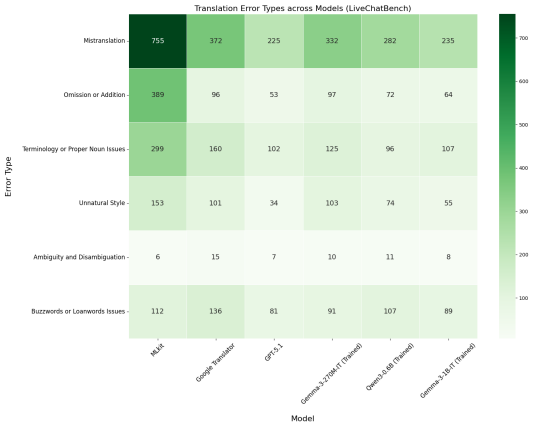


Figure 3: **Heatmap of six types of translation errors.** Darker colors indicate a higher number of errors.

Input: 힘들면 방종해
>>> Ground Truth: If it's tough, end the stream.
>>> GPT-5.1: If it's hard, just let yourself go.
>>> Gemma3-270M (Trained): If it's too hard, end the stream.

Input: 애가 억타인데
>>> Ground Truth: The kid is being forced to play StarCraft, though.
>>> GPT-5.1: The kid is being so 'eokta' though.
>>> Gemma3-270M (Trained): The kid is being forced to play StarCraft.

Figure 4: Comparison of the translation outputs of GPT-5.1 and trained Gemma-3-270M.

serve that on-device models exhibit a substantial performance improvement after training. Indeed, on LiveChatBench, the Qwen3-0.6B shows 158×, 5.29×, and 11× improvements in BLEU, ChrF++, and FSP, respectively. It demonstrates the advantages and potential of on-device models that, with limited resources, can be deployed in real-world applications for well-targeted domain-specific tasks.

When trained for the specific task, on-device models achieve performance comparable to that of commercially available models. In Table 7, We find that, on LiveChatBench, the three on-device models exhibit performance comparable to GPT-5.1 despite their small model sizes. Meanwhile, Figure 3 shows the number of errors recorded for each of the six translation error types only when the severity score exceeded 50. We also observe that, for the six types of translation errors, all three domain-adapted models outperform the Google Translate API and achieve performance close to that of GPT-5.1. However, the results presented in Table 7, Figure 5, and Figure 6 indicate that, on FLORES-200 and WMT24++, the performance of on-device models still falls short of that of commercial models. We assume that the observed performance gap primarily stems from differences in task characteristics, particularly the average sequence length: LiveChatBench has a mean length of 13.98, whereas FLORES-200 and WMT24++ have mean lengths of 65.18 and 97.52, respectively, suggesting that on-device AI still has room for improvement.

Qualitative Analysis. Figure 4 illustrates how domain adaptation helps compact models interpret live-streaming terminology. GPT-5.1 mistranslates "방종" as "let yourself go" and romanizes "억타" as "eokta," whereas the adapted Gemma-3-270M correctly conveys "end the stream" and "being forced to play StarCraft," respectively. These examples suggest that targeted adaptation can enable compact models to capture domain-specific meanings missed by a general-purpose commercial model, despite their limited capacity.

6 Conclusion

In summary, this paper provides empirical evidence on the key considerations when deploying on-device models in mobile environments and on the level of performance that can be expected in practice through extensive experiments. To the best of our knowledge, this is the first practical investigation conducted across diverse mobile device environments, and we believe that our findings provide valuable insights for the on-device AI field.

Limitations

This work has three main limitations, which future research could address. First, **Language Coverage**: Due to budgetary and other constraints, this study conducts experiments exclusively on datasets consisting of Korean–English parallel sentence pairs. Since we did not evaluate the performance of on-device models on chat data that includes meme-related terminology and slang across diverse languages, the generalizability of our findings may be limited. In future work, we should investigate the extent to which domain adaptation is effective for multilingual data.

Second, **Meme and Slang Glossary.** The collected terminology dictionary used to construct the training dataset contains 656 entries, which is insufficient to comprehensively cover all expressions that exist in real-world settings. In future work, we will need to collect a larger number of terms and focus on how to automatically gather newly emerging and disappearing terms on a daily basis and integrate them into our data construction pipeline.

Finally, **Limitations of Resource Measurement in Realistic App Execution.** Although we measured mobile-device resource utilization (Section 4) during translation, the experiments primarily represent standalone translation execution. In practical settings, translation runs concurrently with application logic and OS-level tasks, which may introduce contention and variability in resource allocation (e.g., thermal throttling, background activity, UI thread workload). As a result, our measurements provide an upper-/lower-bound estimate under controlled conditions rather than a definitive characterization of resource usage during real app operation. Future work will conduct end-to-end profiling within a representative third-party livestreaming app scenario and report per-component breakdowns to improve ecological validity.

Ethical Considerations

This study followed established ethical guidelines to ensure the integrity and fairness of all experiments. Data collected from the livestreaming platform were used in accordance with the platform's permissions and contained no personally identifiable information. The Korean-language datasets were used exclusively for academic purposes. All parallel sentence pairs were created through ethically appropriate procedures. The experimental design incorporated strict safeguards for data privacy and protection. All human participants were informed of the study's purpose and procedures and were free to withdraw at any time without penalty. Overall, we sought to conduct AI research that is both scientifically rigorous and ethically responsible, with due respect for privacy, intellectual property, and participant well-being.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model. *Preprint*, arXiv:2502.02737.

Ondrej Bohdal, Konstantinos Theodosiadis, Asterios Mpatziakas, Dimitrios Filippidis, Iro Spyrou, Christos Zonios, Anastasios Drosou, Dimosthenis Ioannidis, Kyenghun Lee, Jijoong Moon, Hyeonmok Ko, Mete Ozay, and Umberto Michieli. 2025. On-device system of compositional multi-tasking in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 416–424, Suzhou (China). Association for Computational Linguistics.

Ona de Gibert, Joseph Attieh, Teemu Vahtola, Mikko Aulamo, Zihao Li, Raúl Vázquez, Tiancheng Hu, and Jörg Tiedemann. 2025. Scaling low-resource MT via synthetic data generation with LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 27662–27680, Suzhou, China. Association for Computational Linguistics.

Daniel Deutsch, Eleftheria Briakou, Isaac Rayburn Caswell, Mara Finkelstein, Rebecca Galor, Juraj Juraska, Geza Kovacs, Alison Lui, Ricardo Rei, Jason Riesa, Shruti Rijhwani, Parker Riley, Elizabeth Salesky, Firas Trabelsi, Stephanie Winkler, Biao Zhang, and Markus Freitag. 2025. WMT24++: Expanding the language coverage of WMT24 to 55 languages & dialects. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 12257–12284, Vienna, Austria. Association for Computational Linguistics.

Tobias Domhan and Dawei Zhu. 2025. Same evaluation, more tokens: On the effect of input length for machine translation evaluation using large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7940–7958, Suzhou, China. Association for Computational Linguistics.

Google. 2019. Translation. *Google for Developers*.

Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A. Horowitz, and William J. Dally. 2016. Eie: efficient inference engine on compressed deep neural network. In *Proceedings of the 43rd International Symposium on Computer Architecture*, ISCA '16, page 243–254. IEEE Press.

Kimberly Hau, Safwat Hassan, and Shurui Zhou. 2025. Llms in mobile apps: Practices, challenges, and opportunities. *Preprint*, arXiv:2502.15908.

Kartik, Sanjana Soni, Anoop Kunchukuttan, Tanmoy Chakraborty, and Md Shad Akhtar. 2024. Synthetic data generation and joint learning for robust code-mixed translation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15480–15492, Torino, Italia. ELRA and ICCL.

Eunsu Kim, Juyoung Suk, Philhoon Oh, Haneul Yoo, James Thorne, and Alice Oh. 2024. CLIcK: A benchmark dataset of cultural and linguistic intelligence in Korean. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 3335–3346, Torino, Italia. ELRA and ICCL.

Yunmeng Li, Jun Suzuki, Makoto Morishita, Kaori Abe, and Kentaro Inui. 2025. MQM-chat: Multidimensional quality metrics for chat translation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3283–3299, Abu Dhabi, UAE. Association for Computational Linguistics.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, Liangzhen Lai, and Vikas Chandra. 2024. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. *Preprint*, arXiv:2402.14905.

Krishna T. Malladi, Frank A. Nothaft, Karthika Periyathambi, Benjamin C. Lee, Christos Kozyrakis, and Mark Horowitz. 2012. Towards energy-proportional datacenter memory with mobile dram. In *2012 39th Annual International Symposium on Computer Architecture (ISCA)*, pages 37–48.

OpenAI. 2025. Gpt-5.1: A smarter, more conversational chatgpt.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Thang M. Pham, Phat T. Nguyen, Seunghyun Yoon, Viet Dac Lai, Franck Dernoncourt, and Trung Bui. 2025. SlimLM: An efficient small language model for on-device document assistance. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 436–447, Vienna, Austria. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Haiyang Shen, Hang Yan, Zhongshi Xing, Mugeng Liu, Yue Li, Zhiyang Chen, Yuxiang Wang, Jiuzheng Wang, and Yun Ma. 2025. Ragsynth: Synthetic data for robust and faithful rag component optimization. *Preprint*, arXiv:2505.10989.

Zhewei Sun, Qian Hu, Rahul Gupta, Richard Zemel, and Yang Xu. 2024. Toward informal language processing: Knowledge of slang in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1683–1701, Mexico City, Mexico. Association for Computational Linguistics.

Zhewei Sun, Richard Zemel, and Yang Xu. 2022. Semantically informed slang interpretation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5213–5231, Seattle, United States. Association for Computational Linguistics.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2022. No language left behind: Scaling human-centered machine translation. *Preprint*, arXiv:2207.04672.

Ifeoluwa Wuraola, Nina Dethlefs, and Daniel Marciniak. 2024. Understanding slang with LLMs: Modelling cross-cultural nuances through paraphrasing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15525–15531, Miami, Florida, USA. Association for Computational Linguistics.

Jiajun Xu, Zhiyuan Li, Wei Chen, Qun Wang, Xin Gao, Qi Cai, and Ziyuan Ling. 2024. On-device language models: A comprehensive review. *Preprint*, arXiv:2409.00088.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Changsheng Zhao, Ernie Chang, Zechun Liu, Chia-Jung Chang, Wei Wen, Chen Lai, Sheng Cao, Yuandong Tian, Raghuraman Krishnamoorthi, Yangyang Shi, and Vikas Chandra. 2025. Mobilellm-r1: Exploring the limits of sub-billion language model reasoners with open training recipes. *Preprint*, arXiv:2509.24945.

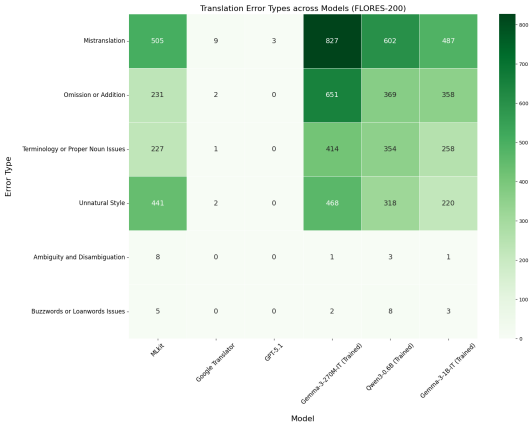


Figure 5: **Heatmap of six types of translation errors (FLORES-200).** Darker colors indicate a higher number of errors.

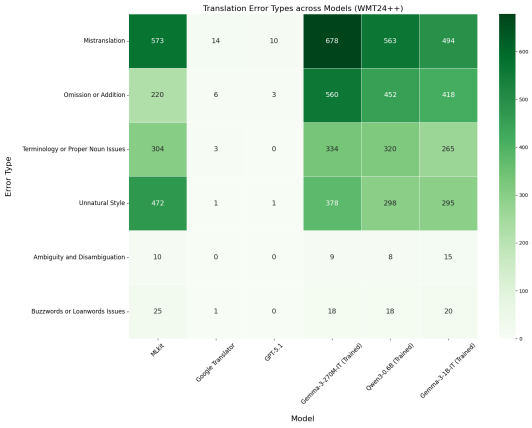


Figure 6: **Heatmap of six types of translation errors (WMT24++).** Darker colors indicate a higher number of errors.

Parameter	Value
Total Rank (r)	64
Scaling Factor (α)	32
Target Modules	`{q, k, v, o, gate, up, down}_proj`
Optimizer	`AdamW`
Warmup Ratio	0.03
Gradient Accumulated Batch	4
Learning Rate	2e-4
Dropout Rate	0.05
Max Sequence Length	256

Table 8: **LoRA training hyperparameters.** We employ a unified configuration for all on-device models.

A Translation Error Distribution

Figure 5 and Figure 6 show the distribution of six translation error types on the FLORES-200 and WMT24++ benchmarks. The three trained on-device models exhibit lower performance than commercial models, which is likely due to the substantial domain mismatch between the live-stream chat data used for training and the domains of the two benchmarks. Indeed, whereas LiveChat-Bench includes meme terminology and slang and has an average length of 13.98, FLORES-200 and WMT24++ do not, with much longer average lengths of 65.18 and 97.52, respectively. These results indicate that there is still room for improvement in on-device AI models, not only in terms of domain adaptation performance but also in achieving better generalization performance.

B Prompt Description

Prompt Template (FSP) for LLM-Based Evaluation of Six Error Types

 # Role:
You are an annotator for the quality of machine translation. Your task is to identify errors and assess the quality of the translation. There may be no translation errors.

 # Instruction:
Based on the source text (in <source></source> tags), machine translation surrounded (in <translation></translation> tags), and information (in <information></information> tags) identify error types in the translation and classify them. There may be no translation errors.

The categories of errors are:

- Mistranslation (Mistranslation refers to fundamental inaccuracies in the translation process, including untranslated source segments incorrect lexical choice or grammar that distorts the meaning, as well as undertranslation and overtranslation.)
- Omission or Addition (Missing source content (omission) or additional content not present in the source (addition) are considered to be Omission or Addition errors.)
- Terminology or Proper Noun Issues (Terminology and Proper Noun Issues are related to inaccuracies when translating specialized vocabulary, inherent terms, and proper nouns from the source text.)
- Unnatural Style (Unnatural Style refers to translations that are grammatically correct but unnatural in the target language.)
- Ambiguity and Disambiguation (Ambiguity and Disambiguation errors occur when the ambiguities or errors in the source text, such as typographical errors, omissions, unclear abbreviations, and erroneous punctuation, are not faithfully reflected in the translation.)
- Buzzwords or Loanwords Issues (Buzzword or Loanword Issues occur when such terms are not translated accurately according to their usage in both the source and target languages. This includes the incorrect translation of popular sayings, newly created words, Internet slang, and memes.)
- other.

Each error is classified as one of three categories: critical, major, and minor.

Critical errors inhibit comprehension of the text. Major errors disrupt the flow, but what the text is trying to say is still understandable. Minor errors are technically errors, but do not disrupt the flow or hinder comprehension.

The source text must be fully covered. Please only include errors and no spans that do not contain errors.

You will be given a full document and its translations, but only score one sentence at a time which is given in in <target_segment></target_segment> tags.

Overall quality score of the translation. After highlighting all errors, please choose the overall quality score.

The quality levels associated with numerical scores:

- 0: No meaning preserved: Nearly all information is lost in the translation.
- 33: Some meaning preserved: Some of the meaning is preserved but significant parts are missing. The narrative is hard to follow due to errors. The text may be phrased in an unnatural/awkward way. Grammar may be poor.
- 66: Most meaning preserved and few grammar mistakes: The translation retains most of the meaning. It may have some grammar mistakes or minor inconsistencies.
- 100: Perfect meaning and grammar: The meaning and grammar of the translation is completely consistent with the source. The text sounds like native text in the target language without any awkward phrases.

Use any number in the range between 0 and 100 for a fine-grained quality score.

Please score the following input
<input>
<source_language>Korean</source_language>
*<source>**[KO]**</source>*
*<target_language>**[EN]**</target_language>*
*<translation>**[TRANSLATED RESULT]**</translation>*
*<information>**[REF]**</information>*

Figure 7: **Prompt used to evaluate six predefined error types with GPT-5.1.** The template follows an FSP (Focus Sentence Prompting) format, providing task instructions, error definitions, and score-range descriptors to guide the model's judgments and output structure.