Multilingual LLM synthetic data generation study 2.0.

Methodology, results and recommendations

Burkert, T.,' Peljak-Łapińska, A., Kwiecień, R. (2026).
Multilingual LLM synthetic data generation study 2.0. TrainAI by RWS.

rws

Acknowledgements

The authors would like to express their gratitude to everyone who contributed to the successful completion of this study.

We thank the professional linguists whose work on data creation and evaluation was foundational to this study.

We are also grateful to David Zelený and Iva Čížková for their project management support, and to Nayna Jaen, Tiffany Small and the RWS marketing team for their help with proofreading, editing, design, and promotion.

Table of Contents

1 About this study

1.1 Motivation

Large language models (LLMs) and other artificial intelligence (AI) models rely on large quantities of data during their training stages. However, getting high quality data, especially in less represented languages, is challenging – not only because the internet is largely Anglocentric, but also because a large portion of publicly available content is either copyrighted, or distributed under licenses that do not allow for commercial use. Furthermore, certain types of content can be very rare or contain personal information, further increasing the need to find new data sources.

One of the main answers to the problem is synthetic data generation, which can "significantly augment or even substitute for real-world datasets, particularly in scenarios where labeled data is scarce, expensive, or sensitive" (arXiv). However, due to the uneven distribution of individual languages in common training datasets, LLMs tend to underperform in less represented languages, diminishing their utility for synthetic data generation.

To find out how well state-of-the-art LLMs perform across various tasks and languages, we generated a large portion of texts in multiple languages, and asked LLMs to perform multiple different tasks. The outputs and the level at which they succeeded were then measured by human expert evaluators.

This study aims to provide human-validated insights into the quality of synthetically generated data in multiple languages and scenarios – information that will be highly valuable to AI practitioners looking to build or upgrade their systems with LLMs for a global audience. It also provides a framework for testing LLMs systematically, as well as practical tips on generating synthetic text data at scale.

1.2 Objectives

When designing this study, we had the following objectives in mind:

- Identify the LLMs offering the most natural and grammatical outputs for specific languages and tasks
- Assess the suitability of individual LLMs for specific languages and tasks
- Compare LLMs' ability to follow instructions
- Compare human-generated text to synthetically generated data
- Validate whether generated data is linguistically diverse
- Provide an indication of the volume of validation/editing required to achieve high-quality datasets from synthetically generated data
- Evaluate the differences in different types and sizes of models
- Measure the tokenizer efficiency of models across languages and the speed of data generation
- Compare results with our previous LLM synthetic data generation study and identify model version improvements

1.3 Changes from previous study

Building on the foundation of our previous study conducted in late 2024, this new iteration refines our methodology while maintaining key parameters (e.g., language scope) to ensure comparability. The following is a summary of the critical updates to our study design.

1.3.1 Updated LLMs

The primary motivation for this iteration of the study is to evaluate the performance of the newest generation of models, specifically assessing improvements in multilingual capabilities.

We selected state-of-the-art models from major providers, including previously tested companies (OpenAI, Anthropic, Google, Meta, Mistral), as well as some new entrants (Alibaba, DeepSeek). Details on model selection are available in Section 1.4.1: Models.

1.3.2 Fewer but more complex tasks

Our previous study tested models on 6 tasks, mostly focusing on single sentences. Results showed relatively minor differences, particularly in sentence generation tasks where scores were consistently high, indicating the evaluation was not challenging enough to effectively differentiate between model capabilities.

Consequently, we retained 2 tasks (conversation generation and text normalization), adapted the translation task to handle short paragraphs rather than single sentences, and introduced a new task: domain-specific paragraph generation. This consolidation focused the evaluation on more complex tasks, increasing general difficulty and reducing the dilution of overall scores with simpler tasks.

1.3.3 Comparison with human-generated data

While the original study focused on comparing LLMs, it also included an experiment based on sourcing comparable data from public datasets with permissible licenses. However, this approach faced scalability issues due to the scarcity of such multilingual data. Furthermore, reliance on translated datasets (like FLoRES) introduced potential quality variances, such as 'translationese.'

For this iteration, we employed our own linguistic resources to create content from scratch for all in-scope tasks, establishing a robust human baseline. The workflow was designed to mimic a realistic scenario comparable to LLM generation: creators worked under time limitations, performed minimal research, and operated without a secondary review or curation step. We utilized 5–7 resources per language to ensure diversity. Our goal was not to produce an idealized, fully fact-checked dataset with flawless grammar, but to establish a realistic baseline of human quality under typical constraints.

1.3.4 Automated data variability measurement

Previously, we used humans to evaluate the variability of generated sentences in batches of 10. While useful, this approach was limited to small batches and subject to cultural interpretation of 'variability.'

In this study, we employed TF-IDF cosine similarity and Type-Token Ratio to measure vocabulary uniqueness and overlap among generated paragraphs. While TF-IDF baselines vary by language, we find this metric more objective and scalable than human evaluation for comparing variance within a single language.

1.3.5 JSON (structured) mode

In 2024, not all major LLMs natively supported JSON Mode for output (sometimes also called Structured Outputs). Therefore, we asked LLMs to generate content in batches using an adapted CSV format. This approach served two purposes: first, to provide structured data for easier parsing; second, to pose an additional challenge for the LLMs, since they needed to follow the requested data schema in the instructions to complete the task. Given that the instructions (as part of the prompt) were provided in the target language, this was a significant challenge, causing many models to fail at generating the data even with multiple retries. In such cases, we ended up imputing data for failed generations with the lowest scores.

This time, we made use of JSON Mode and provided the models with a specific JSON schema in which to return the data. Our goal and expectation were to significantly lower the incidence of failed generations and therefore decrease the need to impute data.

1.3.6 Wider pool of annotators

Addressing previous challenges with initial annotator alignment, we added a dedicated qualification and alignment step to ensure a unified understanding of metrics from the start. We also expanded the pool of resources: rather than three data annotators per language, we used 6–8 annotators, further decreasing potential bias from individual annotators. We continue to utilize professional in-country linguists for all data annotation tasks.

1.3.7 Removed flagging for biases and stereotypes

Our previous study allowed evaluators to flag sentences for potential gender, racial, or ethnic biases, as well as social stereotypes. Given the extremely low incidence of such flags across all models and languages in the previous study, we decided to exclude this categorization from this study.

1.4 Scope

1.4.1 Models

We selected eight models for our study based on the following criteria:

- Performance: High scores in industry-standard benchmarks (such as LMArena or Artificial Analysis Intelligence Index)
- Market leadership: Inclusion of the most capable available models from major LLM providers
- Accessibility: A strategic mixture of proprietary and open-weights models
- Scalability: Inclusion of at least one pair of models of different parameter sizes from the same family
- Capabilities: Representation of both reasoning and non-reasoning models

The following models were tested:

Table 1-1. Eight LLMs

Model name	Provider	Release date	Parameters	Reasoning	Distribution
Claude 4.5 Sonnet	Anthropic	Sep 2025	N/A	Hybrid	Proprietary
DeepSeek V3.1	DeepSeek	Aug 2025	671B	Hybrid	Open-weights
Gemini 2.5 Pro	Google	Jun 2025	N/A	Yes	Proprietary
GPT-5	OpenAI	Aug 2025	N/A	Yes	Proprietary
Llama 4 Maverick	Meta	Apr 2025	400B	No	Open-weights
Mistral Medium 3.1	Mistral	Aug 2025	N/A	No	Proprietary
Mistral Small 3.2	Mistral	Jun 2025	24B	No	Open-weights
Qwen3 235B	Alibaba	Jul 2025	235B	No	Open-weights

1.4.2 Languages

When selecting languages for this study, our intention was to represent a wide spectrum of languages: both from the purely linguistic perspective (language families, writing systems and scripts) and from the training data representation perspective (expected proficiency of LLMs). No consideration was given to specific languages or groups of languages that are supported by the individual models. See Section 3.2.1: Qualification for text normalization for further discussion and analysis of supported languages.

None of the companies behind the tested LLMs provide details on their training data. However, the Common Crawl archives (one of the more prominent sources of data) provides an indication of how well each language might be represented in training datasets. We deliberately included languages sampled from diverse points on the representation spectrum, as the volume and quality of the training data for any given language significantly impacts model performance on that language.

Table 1-2. Eight languages

Language	Language code	Why we included it
English (US)	en	Reference/baseline language, maximum representation
Arabic (MSA)	ar	Arabic language, non-Latin script, right-to-left, medium representation
French	fr	Romance language, high representation
Kinyarwanda	rw	Niger-Congo language, low representation
Polish	pl	Slavic language, medium representation
Tagalog	tl	Austronesian language, low to medium representation
Tamil	ta	Indo-Iranian language, non-Latin script, medium to high representation
Simplified Chinese	zh-CN	Sino-Tibetan language, non-Latin logographic script, high representation

1.4.3 Tasks

The tasks we used to evaluate LLMs' performance are based on real-life multilingual use cases and requests. They are designed to measure both synthetic data generation capabilities (domain-specific paragraph generation and conversation generation) and natural language processing capabilities (text normalization and translation).

Table 1-3. Four tasks

Task	Description	Scope	Metrics
Domain-specific paragraph generation	Create a paragraph of text relevant to a provided domain.	3-5 sentences, 35-45 words	Grammar Naturalness Coherence Specified domain fit
Conversation generation	Create a coherent conversation between 2 speakers based on a provided topic.	10 turns, 50-75 words	Grammar Naturalness Coherence
Text normalization	Normalize the provided sentence based on ASR-related rules for spoken text.	One sentence	Normalization Quality
Translation	Create an accurate translation of a provided paragraph.	3-5 sentences, 35-45 words	Grammar Naturalness Translation accuracy

1.4.4 Volume

For each combination of a model, language and task, we generated 100 samples (of paragraphs, conversations or sentences depending on the task) for evaluation, totaling up to 25,600 samples. Each of the samples was evaluated by 3 professional linguists, resulting in 76,800 annotations. Since each task required 1-4 metrics to be rated, we captured 211,200 ratings in total using standardized Likert scales.

1.5 Challenges and limitations

We identified several methodological challenges, including the following:

- Non-deterministic behavior: LLMs may produce different outputs for the identical prompt
- Input sensitivity: Small changes to inputs (prompts) can impact response quality
- Human subjectivity: Human assessment always introduces a degree of subjectivity and bias

Given these considerations, we acknowledge that this study is not completely reproducible and irrefutable. We accept it is:

- Relevant only for a limited period of time, given the rapid rate of progress in LLM development
- Limited in language scope, since we tested only a representative sample of languages.
- Indicative only, since the tasks tested are unlikely to perfectly match future synthetic data generation needs
- At least partially subjective, given that we intentionally used human reviewers based on the principle that humans are the ultimate arbiters of quality for content created for human consumption.

1.6 Process overview

The data generation and analysis pipeline was structured as follows:

- **Prompt engineering:** A base set of prompts was initially created for each task in English. These were then tested with individual models and iteratively refined to ensure optimal performance for each specific architecture.

- **Topic and subtopic generation:** To ensure diverse content coverage, we generated a list of 10 topics, each containing 10 distinct subtopics, all initially in English.

- **Translation:** Professional linguists translated all prompts, topics, and subtopics into the target languages to ensure linguistic accuracy and cultural relevance before any generation took place.

- **Data generation:** The translated prompts were dynamically populated with the localized topics and subtopics. We then executed the generation scripts for every combination of task, language, and model, storing all outputs alongside their metadata in a centralized database.

- **Data evaluation:** A strict blind review protocol was implemented. Each generated text was individually evaluated by 3 professional linguists using Likert scales. To mitigate bias or annotator drift, data was served in a randomized order, and evaluators were blinded to the source (unaware if the text was human-written or model-generated).

- **Data analysis:** Finally, the raw evaluation data was exported, cleaned, and subjected to statistical analysis to derive insights into model performance and variability.

1.7 Key findings

1.7.1 Multilingual proficiency: the language gap is closing fast

The disparity between LLM performance on well-supported vs. underrepresented languages has narrowed dramatically. Gemini 2.5 Pro achieved scores above 4.5 across multiple tasks in Kinyarwanda – a language in which previous model generations could barely produce coherent text. GPT-5 and Claude Sonnet 4.5 also showed meaningful improvements in long-tail languages. While challenges remain, the current generation of models signals that synthetic data generation and translation are becoming viable for a far broader range of languages than ever before.

1.7.2 Task alignment: one model doesn't fit all

No single model dominated across all tasks and languages. GPT-5 excelled at text normalization and translation but struggled with content generation, particularly in Chinese and Polish. Gemini 2.5 Pro led in conversation generation and translation but was matched by Claude Sonnet 4.5 on domain-specific paragraph generation. Smaller models like Mistral Small 3.2 performed surprisingly well in certain languages while failing in others. Practitioners should expect to evaluate models against their specific use cases rather than relying on overall rankings.

1.7.3 Humans vs. machines: LLMs surpass single-pass human quality

When human creators worked under realistic constraints – limited time, no secondary review, and no extensive research – the top LLMs consistently outscored them across most languages. Humans achieved steady results around 4.5 but did not rank 1st in any language. This doesn't render human expertise obsolete; rather, it suggests a shift in workflows where LLMs produce strong first drafts and humans add value through review, refinement, and specialized judgment.

1.7.4 Cost: thinking harder costs more – tokenizer efficiency matters again

Reasoning models generate up to 10x more tokens during their "thinking" process, amplifying the cost impact of tokenizer efficiency. Gemini 2.5 Pro now leads with 3.67 characters per token – roughly 10% more efficient than GPT-5 and up to 3.5 times more efficient than Claude Sonnet 4.5 on non-Latin scripts. For high-volume or reasoning-intensive workloads, these differences translate directly into significant cost variation, making tokenizer efficiency a renewed consideration when selecting models.

1.7.5 Benchmark drift: today's upgrade might be tomorrow's downgrade

Performance doesn't always improve linearly from one model generation to the next – and weaknesses don't always persist. GPT-5 fell behind smaller models on several content generation tasks where GPT-4o had been competitive. Conversely, Mistral and Llama models closed a 3–4x tokenizer efficiency gap that plagued their predecessors. Perhaps most striking: Llama 3.1 was the slowest model we measured in our previous study, yet Llama 4 Maverick now ranks as the fastest of all tested models. Model upgrades reshuffle strengths and weaknesses unpredictably, reinforcing the need to re-evaluate even familiar model families with each new release.

2 Data creation and generation

2.1 Methodology

2.1.1 LLM data generation

Generating synthetic data at scale presents significant challenges – primarily the need for rigorous prompt variation to prevent outputs from becoming repetitive. We found that one of the most effective mitigation strategies to address this challenge is to provide specific thematic cues, such as distinct domains or topics, to guide the model.

While LLMs are generally capable of generating these taxonomies, whether broad lists or domain-specific subsets, this proficiency is not uniform across all languages. In this section, we detail the specific data generation methodologies and strategies employed in the study.

2.1.1.1 Prompt engineering

Our lead prompt engineer developed a baseline English prompt for each task, adhering to industry best practices. This included segmenting instructions into distinct sections, using clear and unambiguous language for guidelines, and wrapping dynamic inputs in XML-like tags to ensure robust separation of instructions and data.

To enforce structured responses, we utilized strict JSON schemas for every task, applying the same schema across all models to ensure consistency. The full schemas are available in the LLM study GitHub repository.

Following the initial design, we subjected the baseline prompts to an iterative testing phase. Each model was evaluated to verify its ability to produce the expected output. If a model failed to generate valid responses or exhibited obvious compliance issues, the prompt was refined specifically for that model until performance was satisfactory.

Model outputs were assessed against the following criteria:

- **Content quality:** Beyond simple validity, we performed spot-checks (using native speakers or machine translation) to ensure the output was coherent and sounded natural.
- **Length adherence:** We verified that the text produced met the length constraints specified.
- **Schema compliance:** The model demonstrated strict adherence to the required JSON output format.

2.1.1.2 Topic and subtopic generation

We generated an extensive list of topics and subtopics, manually curating it down to 10 distinct topics, each with 10 subtopics. The selection criteria prioritized suitability for a general audience and broad cultural relevance.

Most importantly, we employed professional linguists to translate all input topics and subtopics into the target languages. This step is fundamental to ensuring reliable data generation; if the input prompts are unnatural or poorly localized, the resulting LLM outputs will inevitably degrade.

To illustrate, our topics included categories such as *Books*, *Food & Drink*, and *Movies*, with specific subtopics like *Ernest Hemingway*, *Tiramisu*, and *Star Wars*. We developed two distinct topic sets, recognizing that themes conducive to natural conversation often differ from those best suited for domain-specific paragraph writing. The complete list of topics is available in the LLM study GitHub repository.

2.1.1.3 Prompt and topic translation

All model-specific prompts, topics, and subtopics were translated into all in-scope languages by professional translators. We explicitly instructed the translators to translate faithfully to the source without attempting to 'optimize' or 'engineer' the prompts, as such modifications could compromise the cross-language comparability of the data. All prompts are available in the LLM study GitHub repository.

2.1.1.4 Data generation

Data generation was executed via custom Python scripts managing authentication, inference, retry logic, and database persistence. We utilized the LangChain framework to standardize interactions with model providers, enforce structured output schemas, and capture token usage metadata. Inference was routed through major providers including Azure, AWS, Mistral, and OpenAI (specific provider details are listed in the individual model sections).

The script iterated through every topic-subtopic combination for the given task. These variables were dynamically populated into the prompt template alongside the required JSON schema. For the text normalization task specifically, inputs were passed in batches of 10 sentences.

To mitigate transient network or availability issues, we implemented logic allowing three retries per request. Persistent failures (after three attempts) were logged for analysis, as they typically indicate either severe endpoint instability or the model's inability to generate a valid output. Subsequent re-runs were executed targeting these failed combinations to ensure a complete dataset of 100 outputs per model and task. See individual model sections for details on data generation issues and failures.

2.1.2 Human data creation

Human data creation was designed to mimic the process we used for LLM-based data generation: where possible, we provided both humans and LLMs with the same guidelines and constraints. Humans also created all texts based on topics and subtopics that were previously created and translated as described in Section 2.1.1.2: Topic and subtopic generation and Section 2.1.1.3: Prompt and topic translation. All human data creation work was completed in a bespoke tool.

2.1.2.1 Tooling

For human data creation tasks, our bespoke data collection tool was set up in a way that prevented copying and pasting content. For conversation tasks specifically, the user interface enforced the required structure by presenting 10 distinct input fields, eliminating any potential errors related to turn-counting.

Assignments (domains and topics) were distributed randomly across the human resource pool. To standardize efforts, we established specific target durations for each task and implemented maximum time limits. Furthermore, to guarantee stylistic diversity within the dataset, we capped individual contributions at 20 texts per task and employed a team of 6–7 creators for each language.

Table 2-1. Recommended and maximum task times

Task	Scope	Recommended time (sec)	Maximum time (sec)
Domain-specific paragraph generation	Paragraph	300	600
Conversation generation	10-turn conversation	600	1200
Text normalization	Sentence	60	120
Translation	Paragraph	300	600

2.1.2.2 Selecting creators

For this project, we selected creators with strong linguistic backgrounds, typically translators, teachers, or annotators with formal education in the field. We recruited teams of 5–7 creators per locale. To qualify, all candidates completed a series of tasks designed to verify their understanding of the specific project requirements and rating rubrics.

To ensure reliability, we utilized benchmark items that demonstrated high inter-annotator agreement in our previous study, selecting examples that represented the full spectrum of the Likert scale (where available). During this process, we identified ASR text normalization as a particularly challenging task. Consequently, we introduced an additional batch of targeted qualification data specifically for this workflow (detailed further in Section 3.2.1: Qualification for text normalization) to ensure evaluator alignment.

2.1.2.3 Process

All creators were provided with access to detailed task instructions, which clearly outlined requirements for tone, formality, and length, along with some examples of domain-specific paragraphs and normalization. We explicitly prohibited data creators from using LLMs to generate text or copying content verbatim from existing websites.

While created text was required to be coherent and plausible, we clarified that 100% factual accuracy was not mandatory – we did not expect creators to rigorously fact-check their submissions. The pool of data creators was strictly separated from the annotators to avoid bias and ensure independent evaluation. The maximum number of samples produced per task by one creator was capped at 20 to ensure the final dataset remained diverse.

Our goal with this exercise was not to sample the theoretical best human effort, but rather to simulate a realistic data creation scenario at scale. Although our creators had a linguistic background and passed a qualification step, they worked under time constraints without a secondary review or validation phase. We believe this represents a realistic quality baseline for creating large volumes of varied text data within reasonable budgets.

All instructions shared with the data creators are available in the LLM study GitHub repository.

2.2 Data creation and generation observations

This section outlines notable observations that we gathered during the data creation and generation processes.

2.2.1 Human creation observations

In the human creation phase, we observed notable differences in average creation times across languages. The following diagram illustrates the average duration required to produce a single output, segmented by task and language.

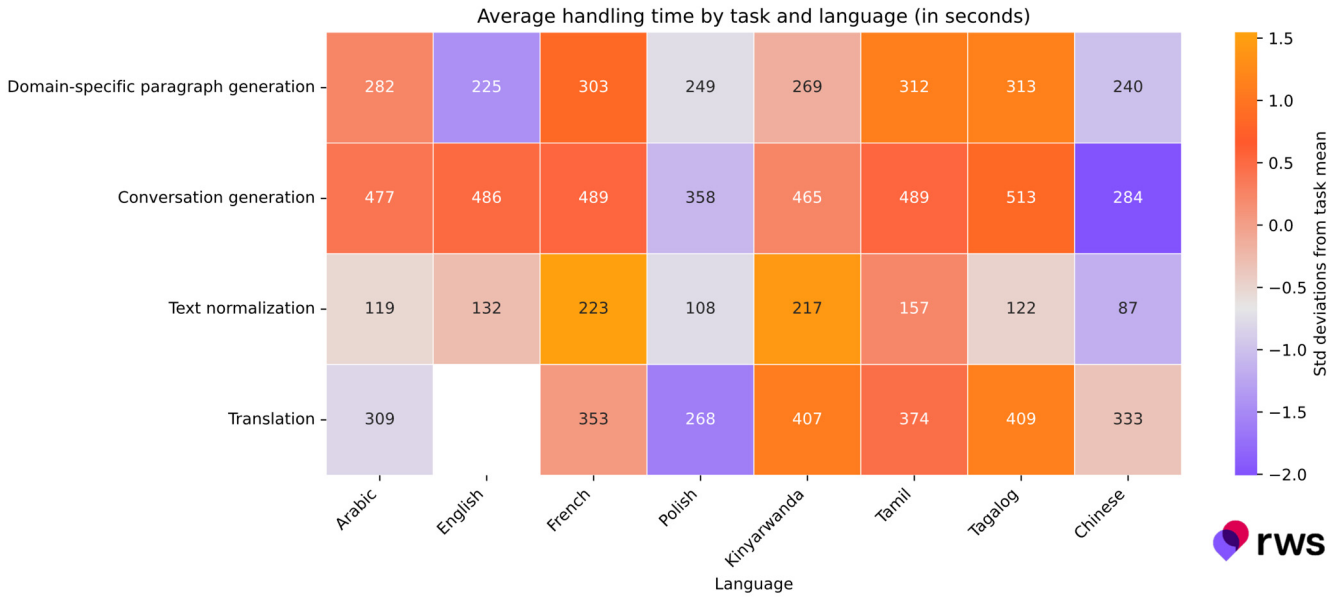

Figure 2-1. Average handling time by task and language (in seconds)

Analysis of average task times reveals that creators in two specific locales, Polish and Chinese, required significantly less time to produce outputs. Both languages exhibited the largest negative standard deviations from the global average. Specifically, Chinese creators were fastest in conversation generation, while Polish creators led in translation speed.

While some of this variance can be attributed to inherent differences in scripts and input methods (particularly in Chinese), we also observed significant fluctuations in completion times among individual creators within the same language.

Overall, the human creation workflow proceeded smoothly. Beyond minor technical glitches and routine clarifications regarding instructions, we encountered no significant operational challenges.

2.2.2 LLM generation observations

Analysis of LLMs' raw generation performance across all 4 tasks revealed significant disparities in adherence to structured output requirements. Schema compliance emerged as a critical differentiator for successful synthetic data generation. Two models, Llama 4 Maverick and DeepSeek V3.1, exhibited particularly severe limitations, repeatedly failing to conform to the specified JSON schema.

Of all LLMs, DeepSeek V3.1 proved the most problematic. It displayed highly unpredictable behavior, generating arbitrary JSON key combinations or, in some instances, outputting the unmodified schema template rather than populated data. Correcting these errors required multiple retry cycles and prompt modifications explicitly designed to reinforce schema adherence.

While Llama 4 Maverick suffered from similar issues, its deviations were more consistent. The model tended to produce 3-4 predictable schema variations, which we were able to manage through hardcoded parsing logic. These findings underscore the critical importance of reliable schema adherence in production pipelines. Models lacking this capability introduce substantial preprocessing overhead and significantly compromise operational reliability.

2.2.2.1 Domain-specific paragraph generation

The domain-specific paragraph generation task revealed an unexpected pattern unique to DeepSeek V3.1, while all other models maintained consistency across languages. Counter-intuitively, domain-specific paragraph generation in French produced the highest rate of schema failures for this model, with English emerging as the second most problematic. This is particularly notable given that English is typically the most reliable language due to its prevalence in training data.

This inverse relationship between expected difficulty and actual failure rates suggests that DeepSeek V3.1 possesses language-specific deficiencies that do not align with typical performance patterns. The successful performance of all other models across target languages indicates that DeepSeek V3.1's failures were model-specific rather than task-inherent.

2.2.2.2 Conversation generation

The conversation generation task proceeded without significant technical difficulties. All models successfully followed the specified JSON schema and generated coherent multi-turn dialogues that met format requirements. This task's success likely stems from its close alignment with the models' pre-training on dialogue data, rendering it less prone to the structural compliance issues observed in transformation-based tasks. Target data volumes were achieved through standard batch processing without requiring prompt modifications or custom parsing solutions.

2.2.2.3 Text normalization

Text normalization tasks revealed critical limitations in how certain models handle structural requirements during linguistic transformations. Both Mistral Medium 3.1 and Mistral Small 3.2 consistently failed to preserve the original target sentence in their outputs, despite successfully normalizing the text. This issue was particularly acute in Mistral Small 3.2, which recorded frequent failures across multiple languages: 21 instances in French, 14 in Arabic, 9 in Tamil, 4 in Kinyarwanda, and isolated cases in English and Polish. Consequently, we were forced to abandon batch processing in favor of processing sentences individually, a workaround that substantially reduced operational efficiency.

DeepSeek V3.1 presented a different challenge. Its inability to adhere to the schema was so severe that it was the only model requiring explicit, redundant instructions within the user prompt to enforce the JSON structure.

2.2.2.4 Translation

The translation task proceeded without significant technical difficulties. All models successfully followed the specified JSON schema and generated translation as per the prompts provided.

3 Data annotation

3.1 Methodology

3.1.1 Tooling

All data annotation work was performed using a bespoke tool with interfaces tailored specifically for individual tasks. Tasks were distributed to annotators randomly and without any indication of the origin of the text (human vs. LLM, or specific models). The only contextual information provided was the task instructions, domain, topic, and the text to be evaluated. To deter the use of external LLMs for rating, the tool disabled copy-pasting functionality.

We employed a replication factor of 3 (each task was annotated by 3 different individuals). To ensure a balanced workload distribution, we capped the number of submissions per single annotator at 700. We utilized 5–7 annotators per language, employing the same selection methodology used for data creators (see Section 2.1.2.2: Selecting creators).

All tasks utilized a 5-point slider for each measured metric (grammar, naturalness, coherence). Certain tasks also included specific metrics, such as specified domain fit for the domain-specific paragraph generation task. The full list of metrics per task is detailed below.

Table 3-1. Task-specific metrics

Task	Metrics
Domain-specific paragraph generation	Grammar, naturalness, coherence, specified domain fit
Conversation generation	Grammar, naturalness, coherence
Text normalization	Normalization quality
Translation	Grammar, naturalness, translation accuracy

Finally, annotators were instructed to flag instances where the text was partially or mostly in the wrong language, which is a common issue for LLM-generated text in low-resource languages. To gain deeper insights into error patterns, we also required annotators to provide short qualitative comments whenever a metric received a low rating (1 or 2).

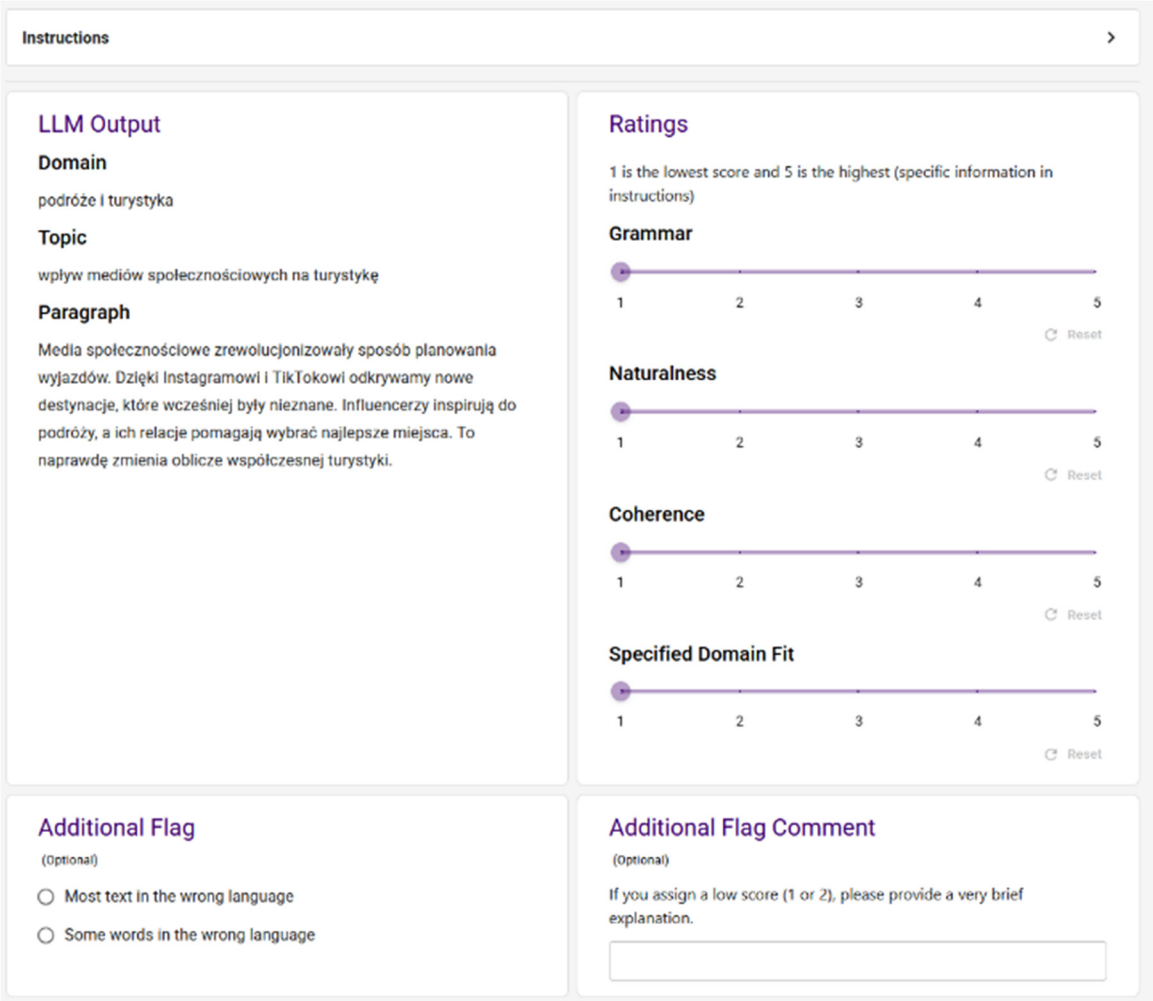

Figure 3-1. Annotation tool user interface

3.1.2 Process

All annotators had direct access to task instructions within the tool. We strictly prohibited the use of LLMs for any part of the workflow and actively monitored for quality anomalies. Key signals included abnormal rating distributions, exceptionally low average handling times, or low agreement with the wider pool (measured via exact agreement, agreement within one point, and Pearson's r correlation).

In a few isolated cases, annotators were removed based on these signals; their submitted ratings were subsequently discarded and re-annotated.

The instructions defined each metric separately, providing detailed criteria for every specific rating level. The following table serves as an example, outlining the rubric for the naturalness metric in the conversation generation task.

Table 3-2. Naturalness rubric: conversation generation

Rating	Description
5	Conversations that read very naturally, with no doubt they were written by a native speaker.
4	Conversations which sound natural but can be improved.
3	Conversations that do not feel entirely natural – they may have been written by a poorly educated person or a non-native speaker.
2	Conversations that mostly do not feel natural – they may have been written by someone who is just starting to learn the language.
1	Conversations that sound completely unnatural.

All instructions shared with the annotators are available in the LLM study GitHub repository.

3.2 Data annotation observations

3.2.1 Qualification for text normalization

Early submissions for text normalization tasks and multiple questions from annotators suggested that a significant portion of the pool did not have a clear understanding of the task. As a result, we designed a 2-step qualification task and gave all annotators feedback on wrongly annotated normalized texts. Only annotators that were able to pass the qualification task were admitted to production on the text normalization task. Annotators' scores improved dramatically after the first feedback round – less than 5% of annotators were offboarded from this task due to failing the qualification.

4 Tokenizer efficiency evaluation

4.1 Methodology

A standard component of our assessment involves measuring the efficiency of model tokenizers in encoding text across different languages. Modern LLMs typically utilize a Byte-Pair Encoding (BPE) tokenizer that breaks down input text into subword units. The tokenizer learns a vocabulary of common character sequences and splits rare words into combinations of these sequences to effectively represent out-of-vocabulary terms. This vocabulary is then frozen for any given LLM (or family of LLMs) and used to translate the input text into a sequence of integer token IDs for processing.

The quality and efficiency of the tokenizer significantly impact inference costs. Inefficient tokenizers, which, on average, encode fewer characters per token, will consume more tokens to generate the same volume of text compared to their more efficient counterparts. This becomes critical at scale, since these LLMs are predominantly accessed via cloud inference APIs that bill based on token consumption.

To measure tokenizer efficiency, we randomly selected 100 texts for each language from the domain-specific paragraph generation task and processed them through each model's tokenizer. This yielded a precise token count for each model, allowing us to benchmark performance on identical data samples for a fair comparison.

4.2 Tokenizer efficiency measurement challenges

During our evaluation of tokenizer efficiency, we encountered persistent issues with content filters in Azure AI Foundry for DeepSeek V3.1 and Llama 4 Maverick. Regardless of the configuration (Default or DefaultV2), we observed frequent false positives flagging content for safety violations or jailbreak attempts.

Consequently, we expanded the evaluation pool to 200 random texts and subsequently selected a subset of 100 that were successfully processed by all models. Given that the objective is to analyze a random sample of plausible texts, we believe this adjustment had no impact on the validity of the tokenizer efficiency results.

However, it is concerning that benign and safe texts in rare languages trigger these filters. This phenomenon presents an additional barrier to the viability of LLMs for low-resource languages.

5 Results analysis

5.1 General model performance

In this section, we present a detailed breakdown of model performance, segmented by task and language. Broadly, Gemini 2.5 Pro established itself as the superior model overall, securing 1st place in 2 of the 4 tasks and ranking a close 2nd in the remaining 2 tasks, with differences in the latter falling within the margin of error. Its results were remarkably consistent across all tasks and languages.

Table 5-1. Top performers across all tasks and languages

Rank	Model name	Average overall score out of 5.0	Minimum overall score out of 5.0	Maximum overall score out of 5.0
1	Gemini 2.5 Pro	4.73	4.56 (rw)	4.85 (fr)
2	Claude Sonnet 4.5	4.61	3.92 (rw)	4.88 (en)
3	DeepSeek V3.1	4.51	3.57 (rw)	4.81 (en)

The other models ranking in the top three – Claude Sonnet 4.5 and DeepSeek V3.1 – remained highly competitive with Gemini 2.5 Pro across most languages, however, they ultimately fell short due to their inability to reliably generate or manipulate text in Kinyarwanda.

Ranking after the top performers is a distinct cluster of 3 models: GPT-5, Llama 4 Maverick, and Mistral Medium 3.1. GPT-5 demonstrated generally high capabilities but suffered from notable instability, recording the lowest score among all models in Chinese.

Qwen3 235B could arguably be included in this competitive tier as well; its performance largely rivaled Llama 4 Maverick, though its overall average was heavily impacted by its severe underperformance in Kinyarwanda. Finally, Mistral Small 3.2 occupied the last position, although it defied expectations by delivering solid scores even in languages beyond English and French.

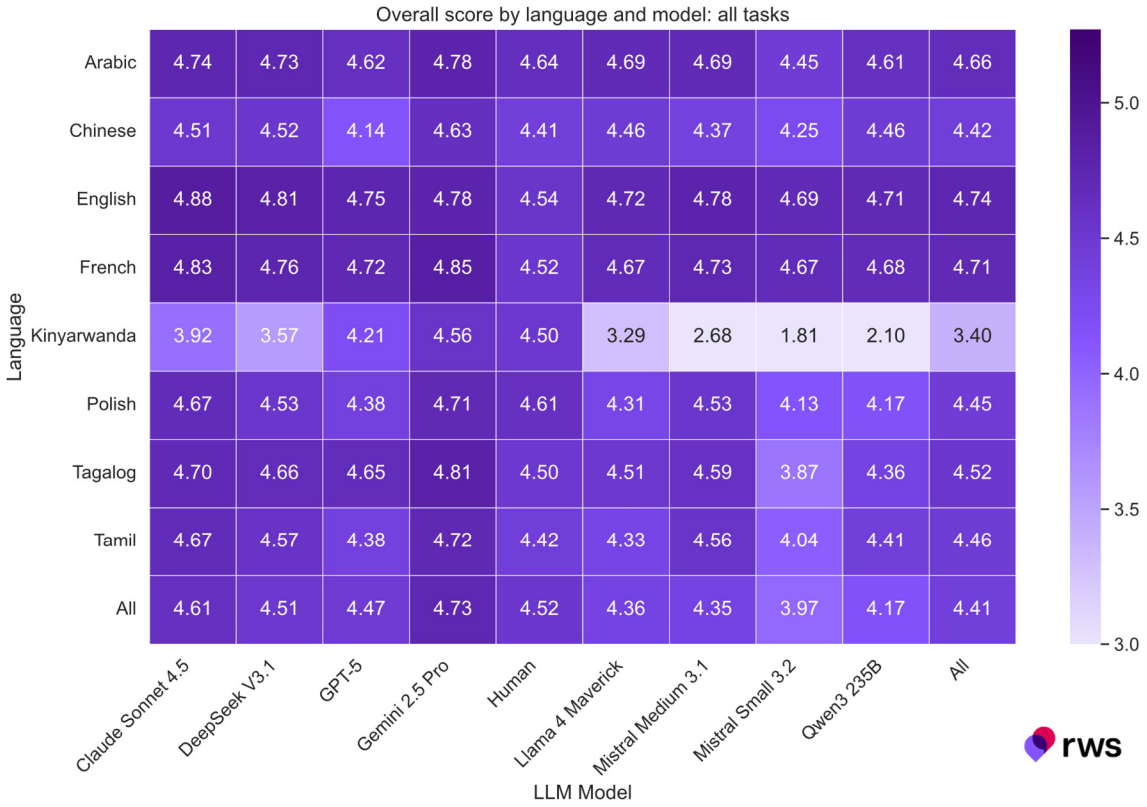

Figure 5-1. Overall score by language and model: all tasks

A high-level analysis of performance by task reveals that most models demonstrated significantly greater proficiency in text generation compared to text manipulation. Two models, however, defied this general trend. Gemini 2.5 Pro stood out for its consistency, maintaining high scores across both domains. GPT-5, on the other hand, displayed an inverse performance profile: while it excelled in text manipulation, its performance on text generation tasks was comparatively poor.

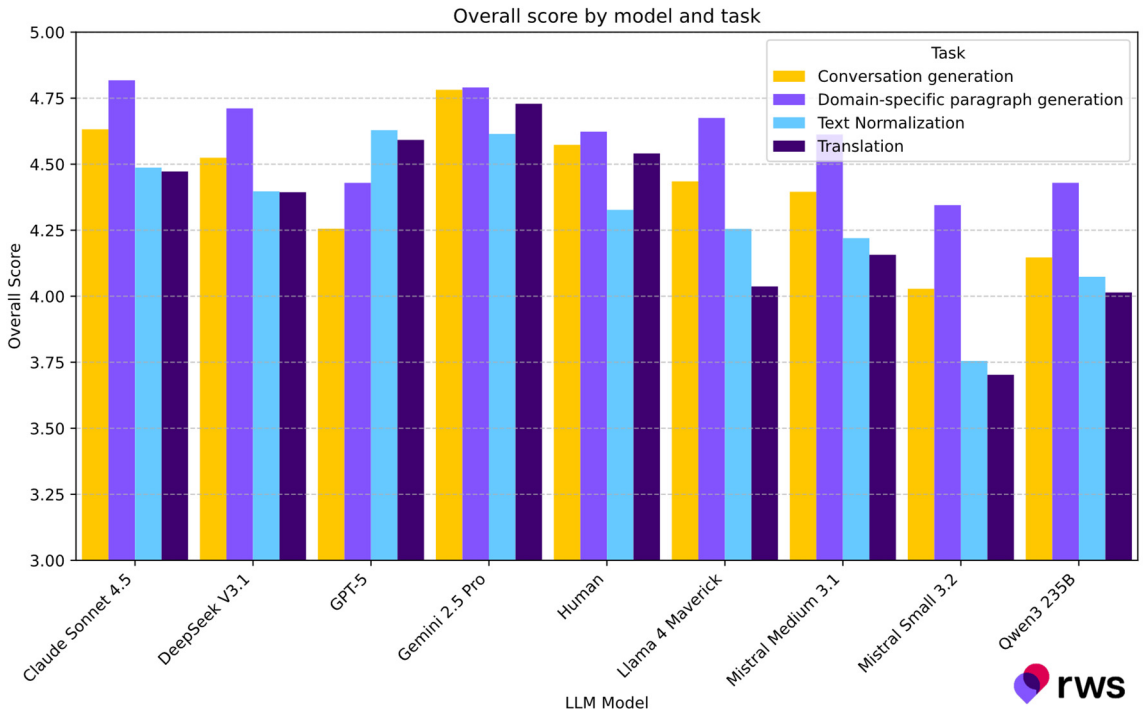

Figure 5-2. Overall score by model and task

5.2 Performance by task

5.2.1 Domain-specific paragraph generation

Table 5-2. Top performers: domain-specific paragraph generation

Rank	Model name	Average overall score out of 5.0	Minimum overall score out of 5.0	Maximum overall score out of 5.0
1	Claude Sonnet 4.5	4.82	4.37 (rw)	4.96 (en)
2	Gemini 2.5 Pro	4.79	4.71 (rw)	4.93 (tl)
3	DeepSeek V3.1	4.71	3.97 (rw)	4.92 (fr)

5.2.1.1 General observations

Domain-specific paragraph generation saw very close results from the top 3 models, with only a 0.03 difference in the overall score between the 1st and 2nd place. The difference is within the margin of error (±0.034), suggesting that both models scored equally well. While all the models scored well (> 4.5) for English, French, and Tagalog, we observed significant differences in other languages. We will now examine notable outliers from our initial expectations.

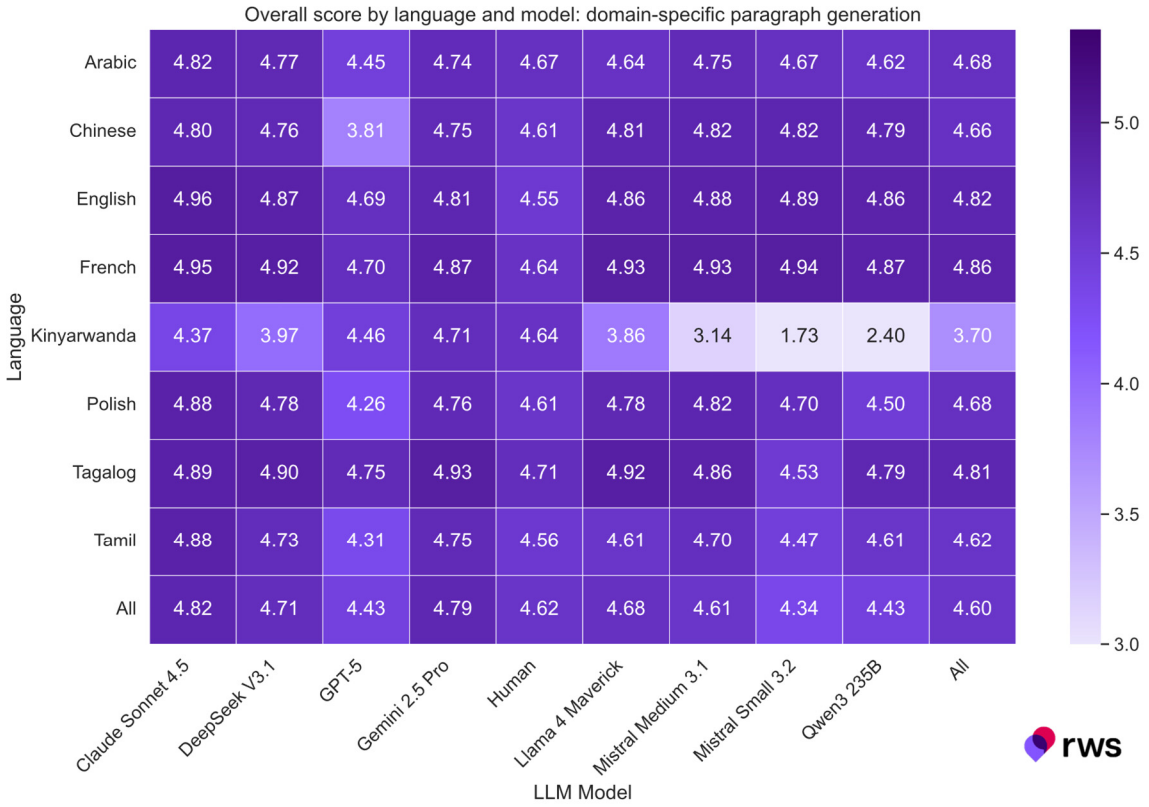

Figure 5-3. Overall score by language and model: domain-specific paragraph generation

The most difficult metric overall was naturalness, in which many models struggled to reach the 4.5 threshold for high-quality content. Regarding outliers, the most surprising result was the very high score of 4.60 that Gemini 2.5 Pro achieved on Kinyarwanda – a language that previous model generations were unable to reliably generate as per our first study. GPT-5 and Claude Sonnet 4.5 also scored well on Kinyarwanda (4.46 and 4.31 respectively), indicating that the wider language coverage is not entirely isolated to Google's models. However, Kinyarwanda was one of the few languages where GPT-5's performance did not contradict expectations, since it scored lower than even its smaller counterparts in most of the languages we tested.

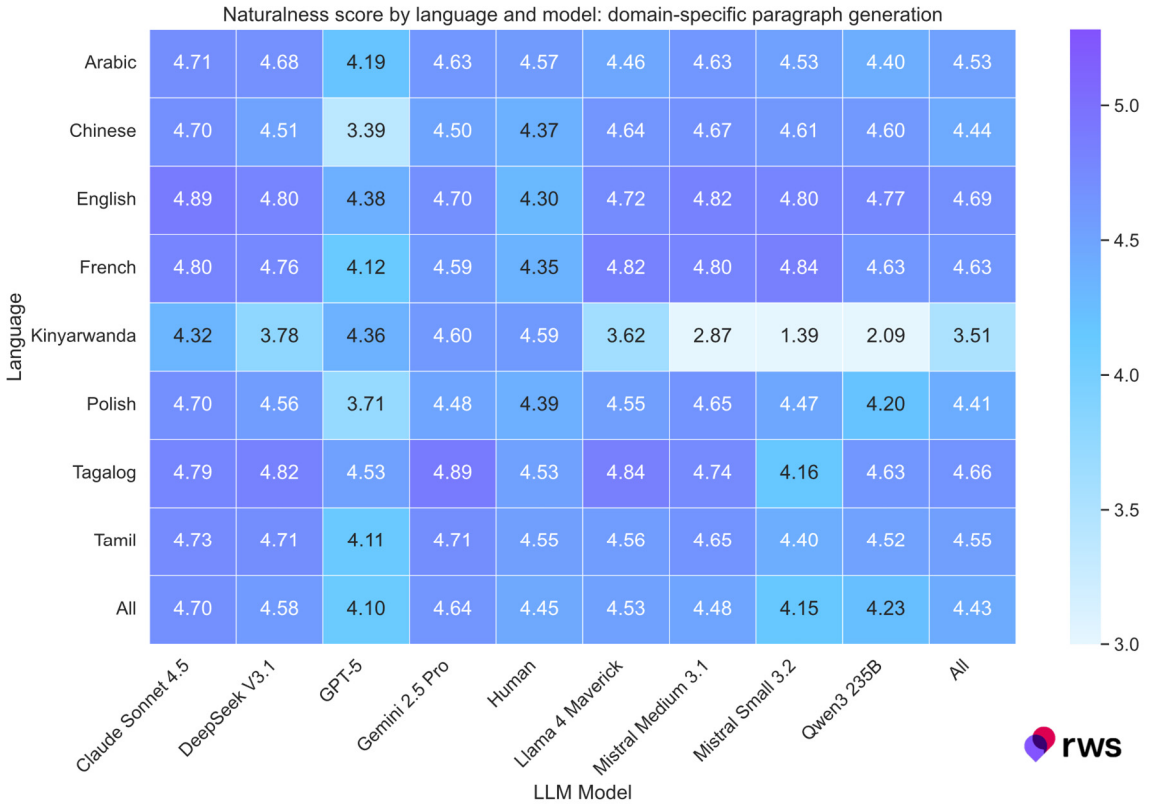

Figure 5-4. Naturalness score by language and model: domain-specific paragraph generation

We also evaluated specified domain fit by measuring the relevance of generated paragraphs to the requested domain and topic. Most of the scores were close to 5.0, with the exception of languages where individual models struggled with grammar or naturalness. Our results indicate that if the model is relatively fluent in a language, generating relevant content is not problematic.

The coherence metric did not yield any notable results, since it highly correlated with naturalness and grammar. We did not find instances of models producing natural and grammatical content with low coherence.

5.2.1.2 GPT-5's underwhelming performance

Notably, GPT-5's performance on Simplified Chinese was far below expectations, with a score of only 3.81 compared to 4.75 for the next-lowest model. Evaluators frequently noted that GPT-5 inserted unnecessary spaces or used spaces instead of punctuation, making the text difficult to read. Due to the frequency of missing punctuation, more than 30% of GPT-5's generated paragraphs received low grammar scores (1 or 2).

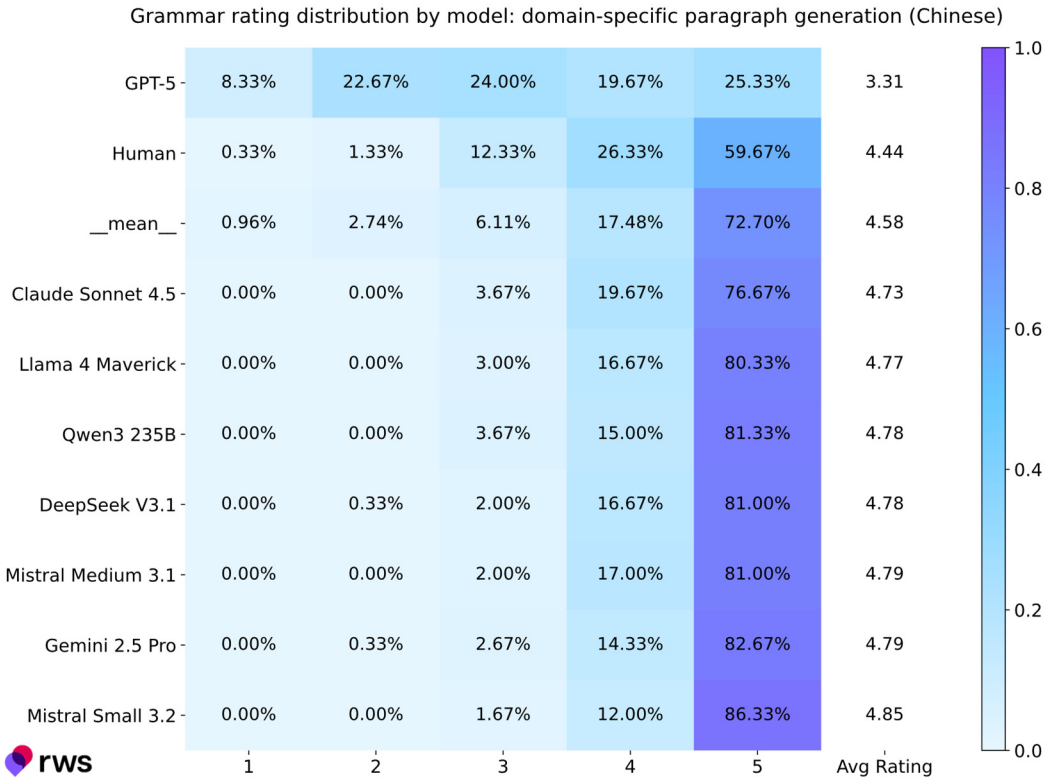

Figure 5-5. Grammar rating distribution by model: domain-specific paragraph generation (Chinese)

Similarly, GPT-5's performance on Polish was poor, resulting in an overall score of 4.26. Feedback from Polish evaluators cited multiple instances of suspected "translationese," i.e. text that follows the source syntax and style too literally, compromising fluency. However, since there was no source text in this task (all models were prompted in the target language), this indicates a fundamental generation issue. Some sentences were often deemed clumsy or unnatural, resulting in almost 17% of the ratings receiving a low score (1 or 2). No other models exhibited this specific pattern in Polish.

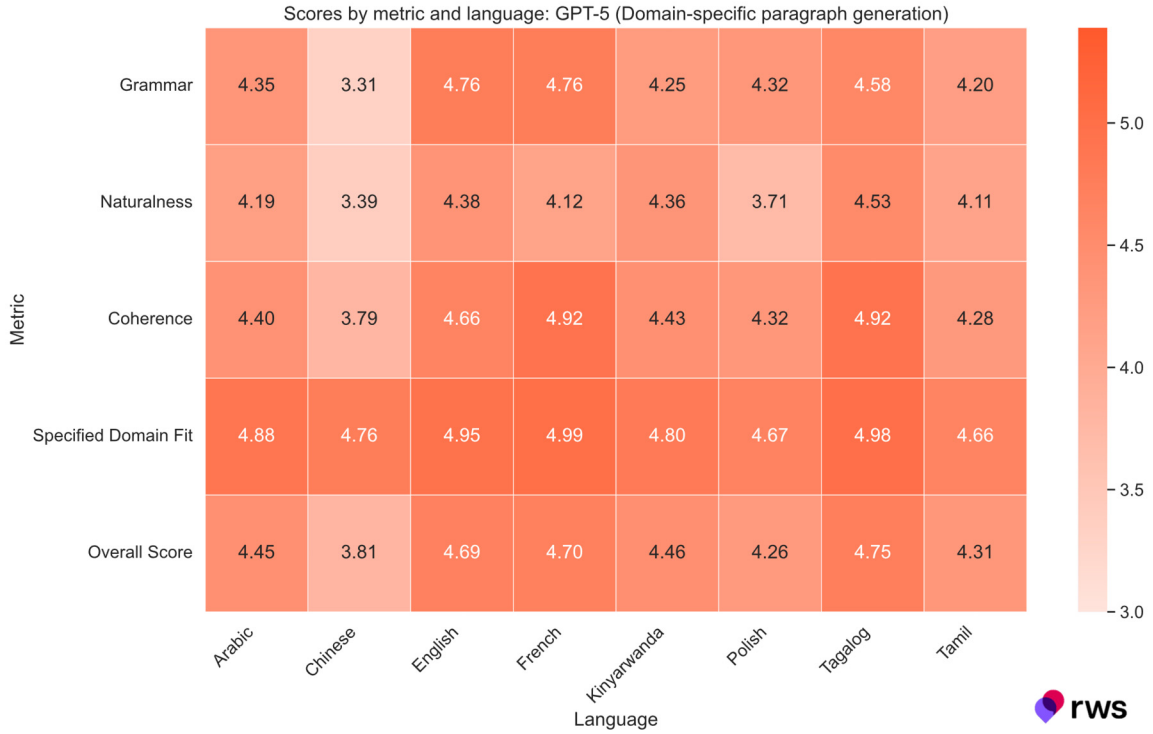

Figure 5-6. Scores by metric and language: GPT5 (domain-specific paragraph generation)

GPT-5 also struggled with Tamil, finishing in last place with an overall score of 4.31. While 4.31 would not be considered a low overall score in isolation on a complex language like Tamil, it is noteworthy that GPT-5 was surpassed not only by other SOTA competitors like Claude Sonnet 4.5 or Gemini 2.5 Pro, but also by the smaller (24B) open-weights Mistral Small 3.2 model, which scored 4.47.

5.2.1.3 Solid scores from Mistral and Llama

While Mistral models and Llama Maverick did not take the top spots overall, they achieved SOTA performance in a large majority of languages. Llama 4 Maverick and Mistral Medium 3.1 came within 0.1 points of the top score in 6 languages. On some occasions, they even beat the overall champion, Claude Sonnet (e.g., Llama on Tagalog, Mistral on Chinese). In more common languages, their usability for generating synthetic domain-specific content was excellent.

Mistral Small 3.2 also demonstrated that small language models now cover a wider scope of languages than prior generations. In our previous study, small models were only usable on major languages such as French or English. However, Mistral Small 3.2 achieved high overall scores (> 4.5) on this task in 6 out of 8 languages tested; its overall performance was primarily impacted by a very low score on Kinyarwanda.

5.2.1.4 Model recommendations: domain-specific paragraph generation

For tasks like domain-specific paragraph generation, we recommend Claude Sonnet 4.5 and Gemini 2.5 Pro as primary choices. However, despite Claude Sonnet 4.5's victory overall, its usability for long-tail languages appears to be lower than that of Gemini 2.5 Pro. Gemini Pro's performance was highly consistent across all languages tested, with even its lowest score (4.71 on Kinyarwanda) representing an excellent result.

For use cases that do not require a wide language scope, DeepSeek V3.1, Llama 4 Maverick, and Mistral Medium 3.1 are also strong alternatives.

5.2.2 Conversation generation

Table 5-3. Top performers: conversation generation

Rank	Model name	Average overall score out of 5.0	Minimum overall score out of 5.0	Maximum overall score out of 5.0
1	Gemini 2.5 Pro	4.78	4.53 (rw)	4.92 (en)
2	Claude Sonnet 4.5	4.63	3.59 (rw)	4.97 (en)
3	DeepSeek V3.1	4.52	3.36 (rw)	4.94 (en)

5.2.2.1 General observations

The conversation generation task was more difficult for the models to complete compared to the domain-specific paragraph generation task: the aggregated score for all models and languages dropped from 4.60 to 4.42, and we saw a much higher incidence of overall scores below 4.0. The score difference among the top 3 models also grew significantly, widening from 0.03 to 0.26. The margin of error was calculated at ±0.037.

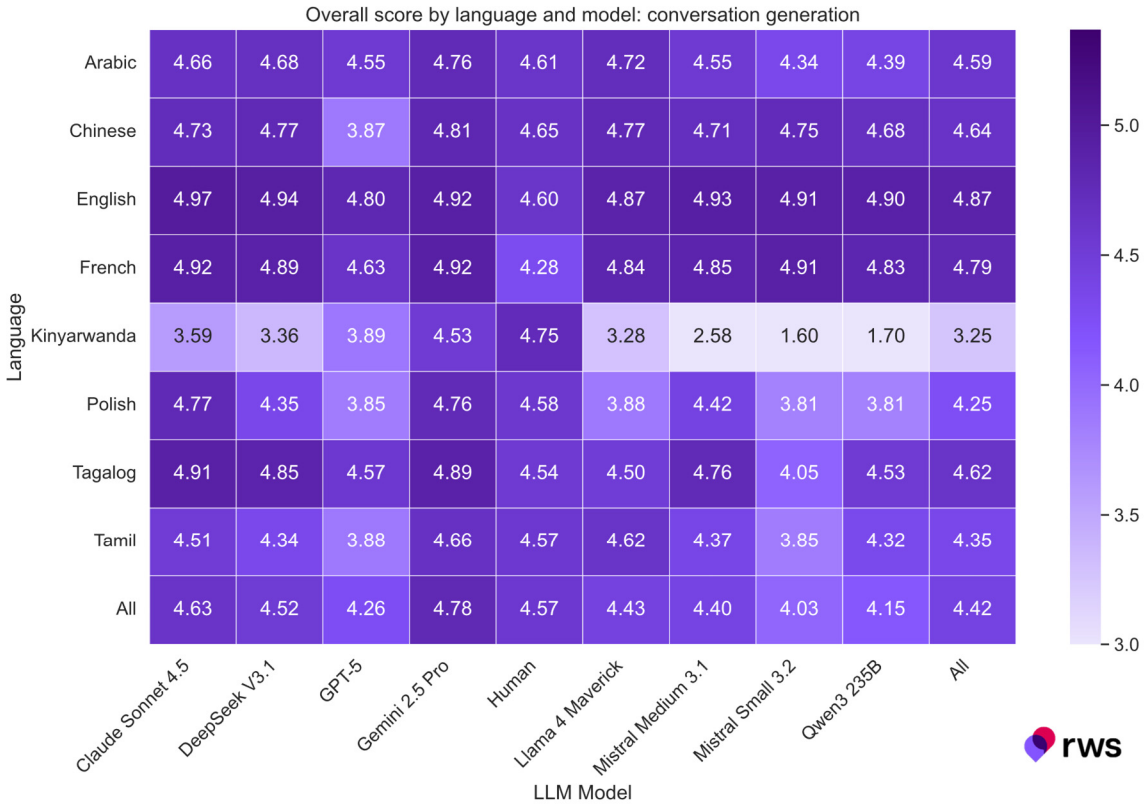

Figure 5-7. Overall score by language and model: conversation generation

One reason for the lower score results is the structural difference between tasks: conversation generation does not include specified domain fit, a generally high-score metric that effectively raised average scores for domain-specific paragraph generation. However, the decline persists even when we isolate the naturalness metric: we observed an average decrease of 0.12 points, widening to 0.28 for the lowest-performing models.

This trend is not driven solely by low-resource languages either. For instance, Llama 4 Maverick's and Mistral Small 3.2's naturalness scores on Polish fell by almost a full point between the two tasks.

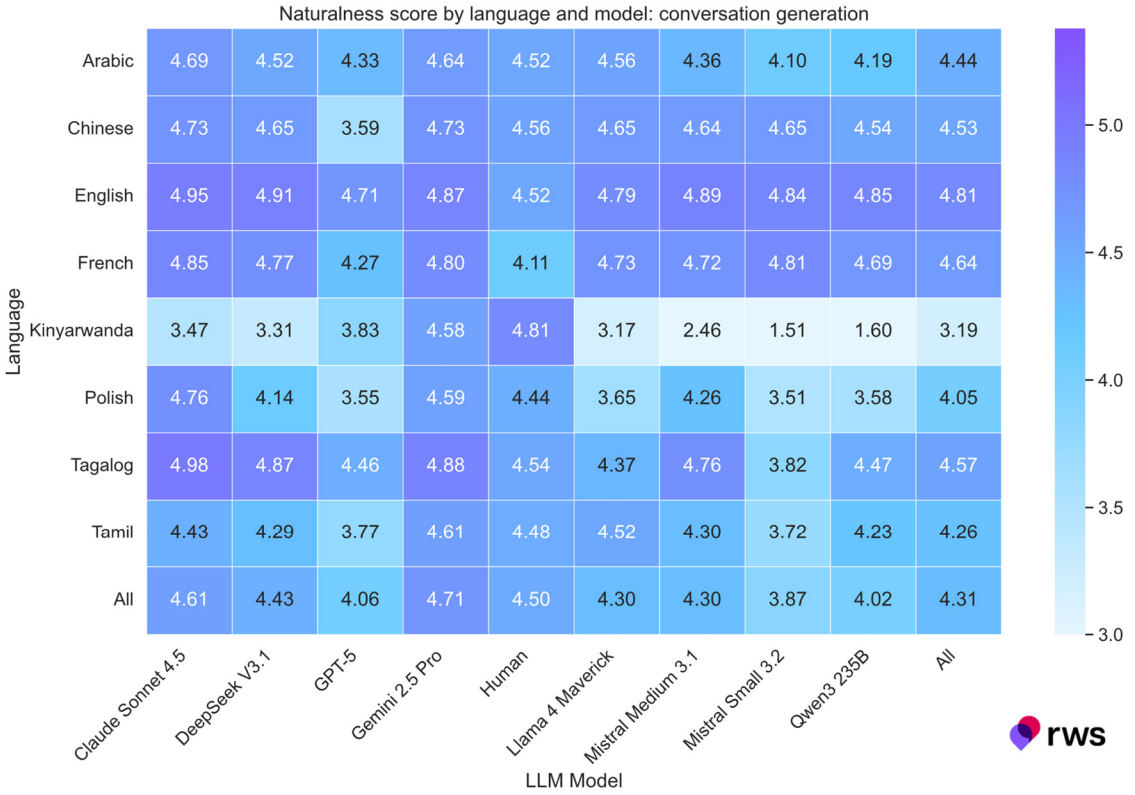

Figure 5-8. Naturalness score by language and model: conversation generation

Like the previous domain-specific paragraph generation task, the coherence metric for conversation generation did not yield any notable results: it correlated highly with naturalness and grammar. We did not find any instances of models producing natural and grammatical content that lacked coherence.

5.2.2.2 Adherence to task instructions: conversation generation

The conversation generation task imposed two basic constraints: 10 conversation turns and a total length of 70–90 words. We intentionally did not specify whether speaker 1 and speaker 2 labels should count towards that limit, but the implicit expectation was that they should not.

Three models frequently failed to adhere to the 10-turn requirement: DeepSeek V3.1 (4 times), Mistral Small 3.2 (5 times) and Qwen 3 235B (5 times). No clear pattern emerged in terms of language incidence: DeepSeek V3.1 failed on Polish and French, while Mistral Small 3.2 failed on Arabic and Tagalog.

We also analyzed the number of words produced by each model and a notable pattern emerged. Due to an ambiguous French translation of the prompt, models occasionally interpreted the instruction as requiring individual conversation turns of up to 90 words. Gemini 2.5 Pro produced an average of 442 words per conversation. Notably, this excessive length did not impact its output quality: it ranked 2nd in naturalness (0.01 points from the top score), 3rd in grammar (0.02 points from the top score), and 1st in coherence in French. Mistral Medium 3.1 also exceeded the limit significantly with an average of 220 words per conversation, while maintaining very high scores.

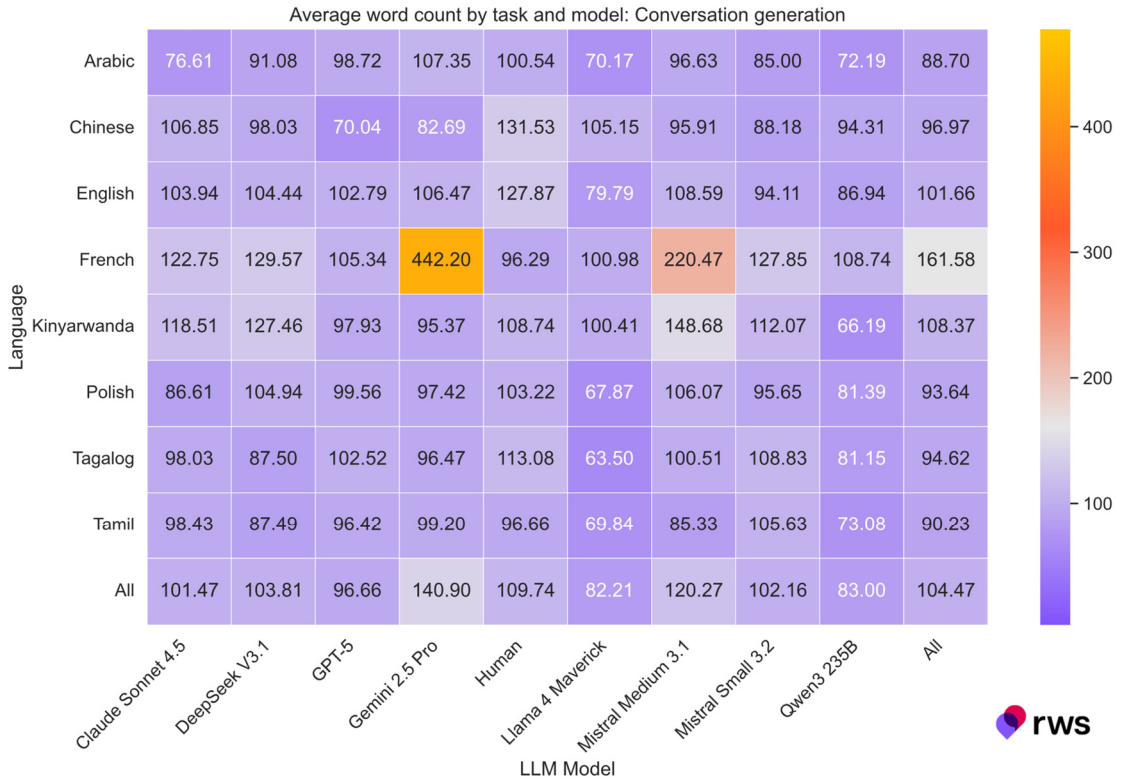

Figure 5-9.Average word count by language and model: conversation generation

A more concerning pattern was the very short conversations produced. On average Llama 4 Maverick, in particular, scored below the lower limit of 70 words in 3 languages. If you consider that 20 of those words were speaker labels, 2 additional languages would fall below the limit when labels are deducted. It was also the only model which had at least one conversation that fell below 70 words in every language.

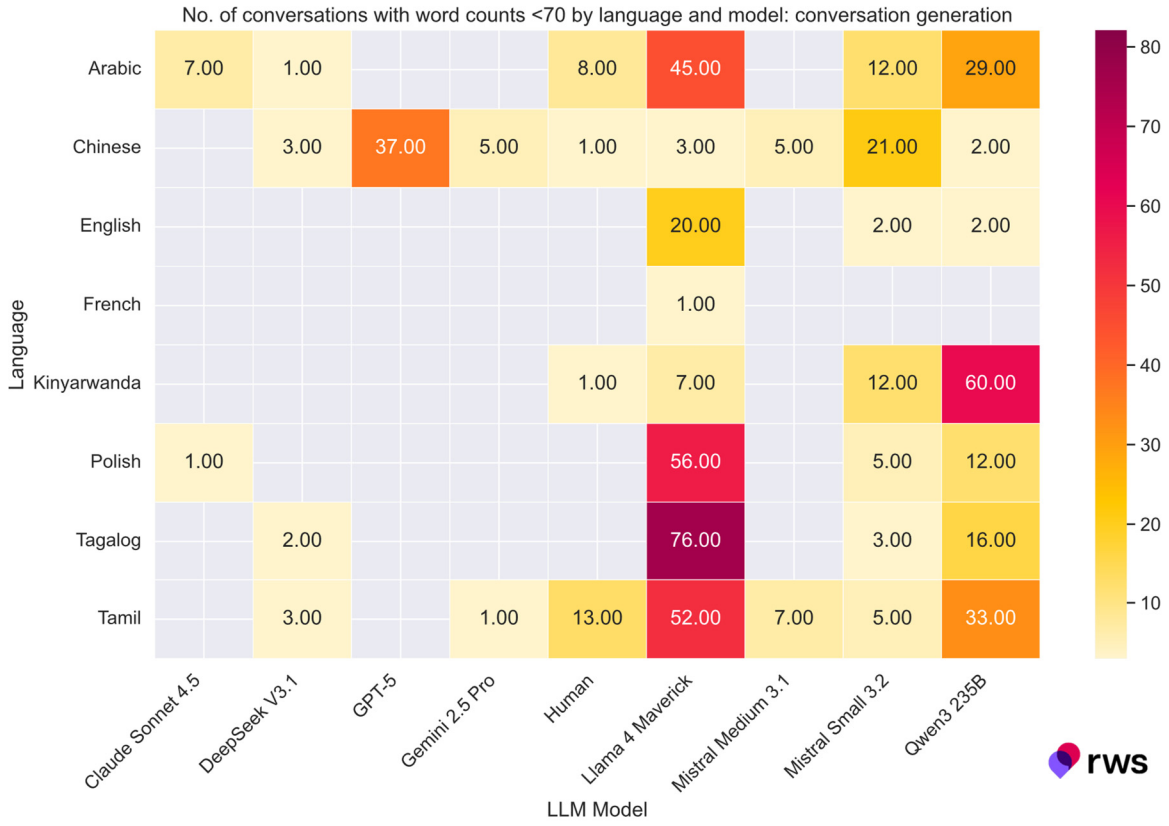

Figure 5-10. No. of conversations with word counts < 70 by language and model: conversation generation

Qwen 3 235B also produced shorter conversations in Kinyarwanda, but given its overall score of 1.70, we can conclude that the model cannot produce meaningful text in that language regardless of length constraints.

We argued in our first study that, due to the different semantic or informational loads of an average word across languages, defining requirements by word count is not effective and can lead to issues. This applies not only to languages that do not use natural word delimiters (such as Chinese, Japanese, or Korean), but also to agglutinative languages or languages that make heavy use of compound words.

However, there do not seem to be any viable alternatives, as sentences tend to have significantly varied lengths and character count is not a good proxy for semantic load when comparing different languages. Therefore, the ability of LLMs to produce texts that conform to specific word count ranges remains important. Despite limitations stemming from the tokenized representation of text, most models were able to adhere to our word count requirements when given a reasonable range. We recommend the range-based approach, as our internal tests have shown that requiring a precise word count limits LLMs' options and can lead to a loss in output quality.

5.2.2.3 GPT-5's underwhelming performance

Conversation generation proved to be another area where GPT-5 fell short of expectations. While it showed relatively competitive results in English, Arabic and Tagalog, its scores on Simplified Chinese, Polish and Tamil were among the lowest of all models tested.

There was no clear pattern regarding the root cause for its lack of performance in Simplified Chinese; evaluator feedback primarily cited disconnected or illogical conversation turns, resulting in a low coherence score of 4.16.

Polish evaluators found a wider variety of issues, including gender mismatches, incorrect grammatical cases, and a lack of quotation marks around proper names (e.g., book or movie titles). Notably, however, feedback echoed the disconnectedness observed in Chinese. Evaluators remarked that speakers "seem not to really talk to one another," or that they make isolated statements rather than responding to the previous turn.

Tamil evaluators similarly called out grammar issues and poor fluency, alongside occasional disconnectedness ("Some utterances not following the previous utterance").

As a result of these structural and grammatical issues, only 50-60% of conversations generated in these three languages received high scores in grammar and naturalness.

Finally, GPT-5 showed mixed results on French. While it produced grammatical conversations (scoring 4.87), it struggled to make them sound natural (scoring 4.08). Evaluators flagged multiple instances of Anglicisms, i.e. English words not naturally used in French, such as "deal" or "chunks," as well as unnatural lexical choices.

Echoing Polish feedback from the domain-specific paragraph generation task regarding "translationese," a French evaluator noted that "the translation is too literal to really make sense in French." This suggests a broader issue: even in tasks where no translation is occurring, text generated in non-English languages appears heavily influenced by English syntax and vocabulary.

5.2.2.4 Gemini 2.5 Pro speaks Kinyarwanda

We previously noted that some models (namely GPT-5, Claude Sonnet 4.5 and Gemini 2.5 Pro) were surprisingly proficient at the domain-specific paragraph generation task. While those results did not carry over to the conversation generation task for GPT-5 and Claude Sonnet 4.5, we measured a very high overall score of 4.53 for Gemini 2.5 Pro. Notably, it was also the only model that scored above the 4.0 threshold on Kinyarwanda.

A key finding in our first study was that LLMs are not ready for long-tail languages, but Gemini 2.5 Pro sends a strong signal that this is rapidly changing. Analyzing individual metrics, the model scored 4.58 in naturalness and 4.30 in grammar, confirming its high proficiency in Kinyarwanda. Only 2% of the generated conversations received low naturalness scores (with similar results for grammar), making the model highly usable for similar use cases.

While we tested a limited scope of languages, Gemini 2.5 Pro's consistent performance suggests high proficiency across a vast array of languages, including long tail and underrepresented ones.

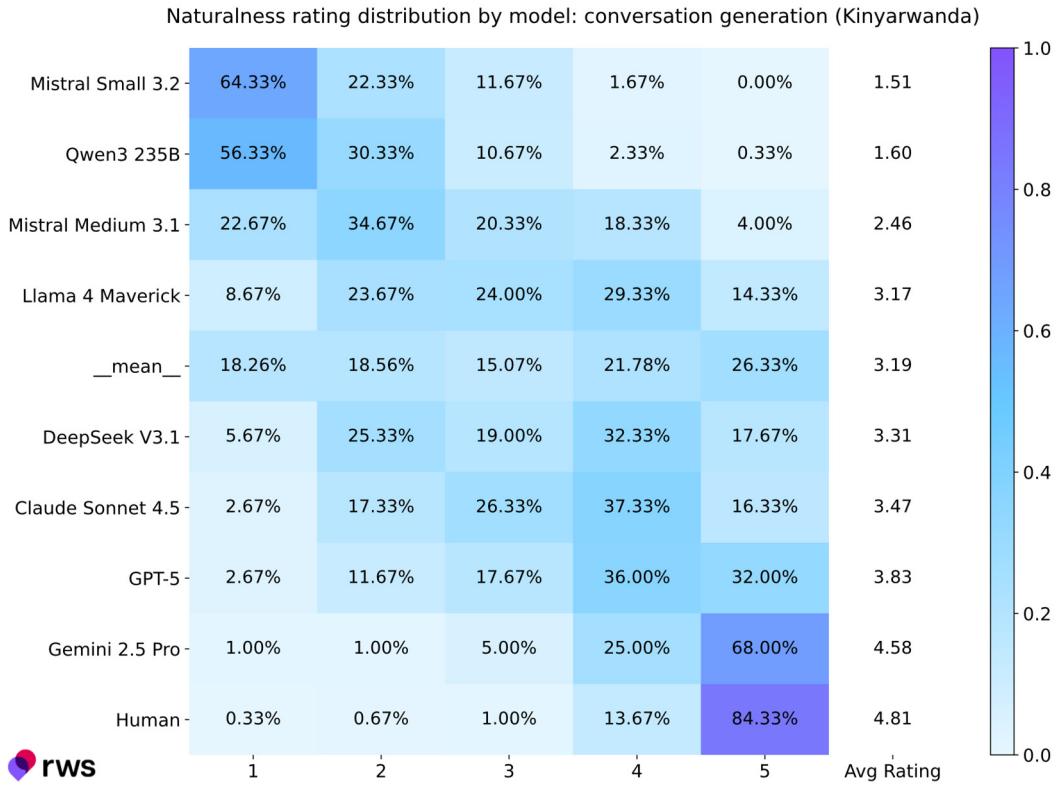

Figure 5-11. Naturalness rating distribution by model: conversation generation (Kinyarwanda)

5.2.2.5 Polish still a challenge for some LLMs

Even though some models achieved very high scores on Polish (Claude Sonnet 4.5 and Gemini 2.5 Pro scored 4.77 and 4.76 respectively), many others struggled with the conversation generation task. Half of the tested models (4 out of 8) scored below 4.0, notably including GPT-5 and Llama 4 Maverick.

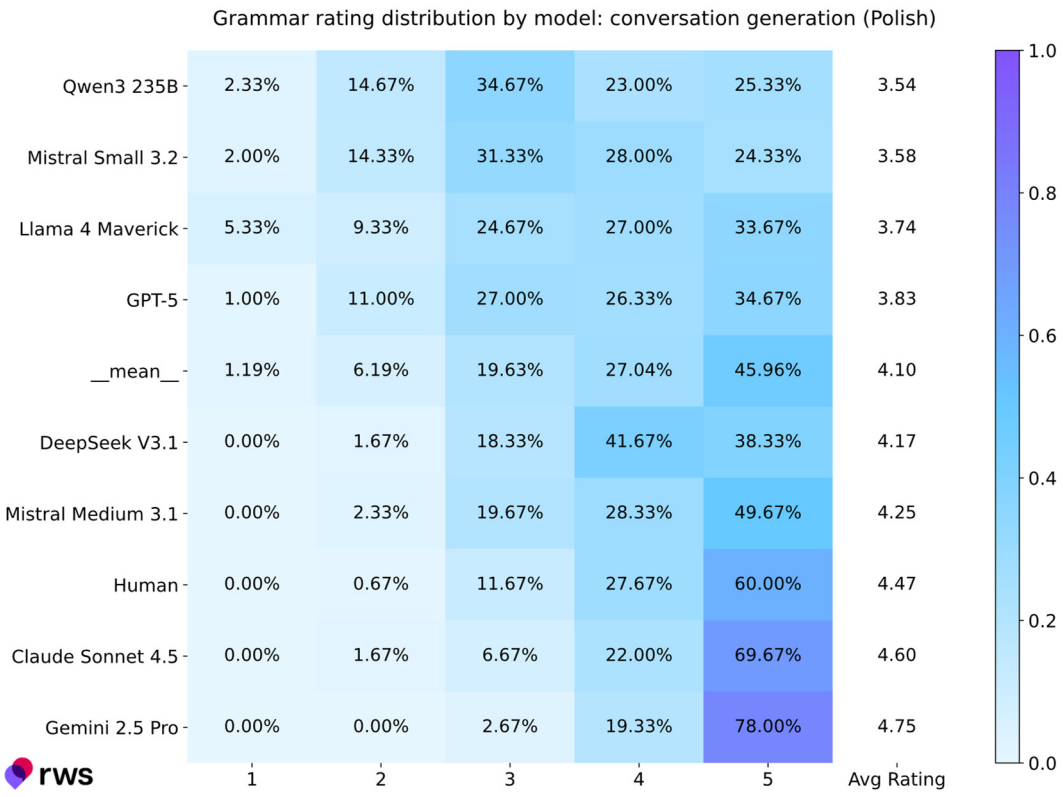

Figure 5-12. Grammar rating distribution by model: conversation generation (Polish)

The most common issues with conversation generation in Polish were punctuation and grammar, specifically the use of English straight quotation marks instead of Polish curly ones, incorrect grammatical cases, and gender inconsistencies.

Language adherence was also a significant issue. Llama 4 Maverick generated 2 conversations entirely in English and occasionally produced text with incorrect special characters (diacritics). Mistral Small 3.2 occasionally reverted to English for individual conversation turns, and Qwen 3 235B used English or hallucinated words.

5.2.2.6 Model recommendations: conversation generation

Given its distinct lead in overall performance, Gemini 2.5 Pro is our primary recommendation for conversation generation. As the overall winner on this task, it ranked 1st in 5 out of 8 languages and showed unmatched consistency, scoring over 4.5 in all languages (and over 4.75 in 6 out of 8 languages).

Claude Sonnet 4.5 was also highly competitive in most languages but fell short in Kinyarwanda and Tamil. Consequently, for highly multilingual conversation generation use cases that may require long-tail language support, we exclusively recommend Gemini 2.5 Pro.

For multilingual use cases where long-tail languages are not required, Claude Sonnet 4.5, DeepSeek V3.1, and Mistral Medium 3.1 are viable alternatives.

5.2.3 Text normalization

Table 5-4. Top performers: text normalization

Rank	Model name	Average overall score out of 5.0	Minimum overall score out of 5.0	Maximum overall score out of 5.0
1	GPT-5	4.63	4.07 (rw)	4.81 (fr)
2	Gemini 2.5 Pro	4.61	4.34 (rw)	4.83 (ar)
3	Claude Sonnet 4.5	4.49	3.91 (rw)	4.79 (ar)

5.2.3.1 General observations

Text normalization is categorically a different task from the previous two we tested: instead of creating text from scratch, models take provided text and make changes to it. We observed a surprising comeback on this task from GPT-5, which scored 4.63 overall and ranked 1st, even though its lead to the runner-up Gemini 2.5 Pro was only 0.02 points (well within the margin of error at ±0.042).

Since this task only consisted of a single rating (normalization quality), the overall score also represents the scoring for normalization quality.

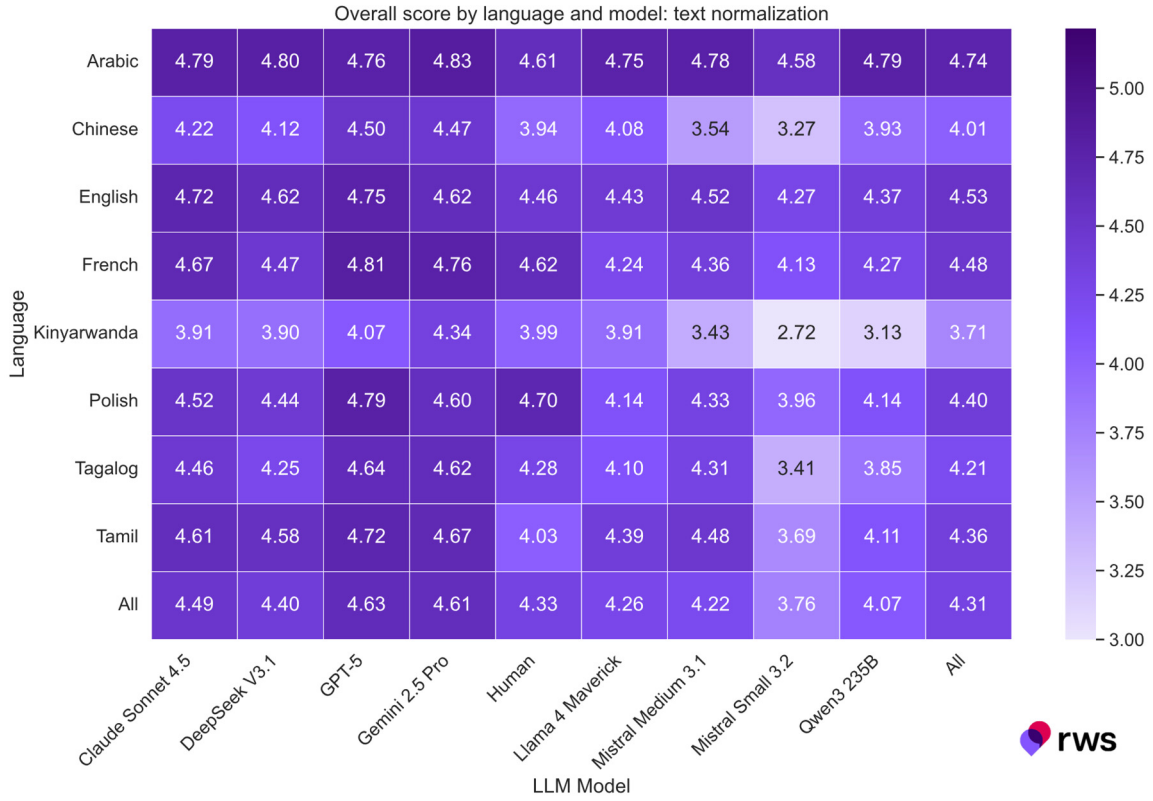

Figure 5-13. Overall score by language and model: text normalization

5.2.3.2 Deceptively strong results on Arabic

Scores on Arabic across all models including small models like Mistral Small 3.2 are significantly higher than other languages, including well-supported languages like English or French. This anomaly is explained by the fact that Arabic naturally normalizes abbreviations and acronyms, making the task considerably easier to complete in Arabic compared to other languages. For example, the acronym FIFA is translated to Arabic as الفيفا, which could be transliterated as *alfifa*. For additional context and examples, see A Comparative Analysis of Abbreviations in Arabic and English.

As a result of the specifics of Arabic morphology, the number of entities requiring a change while normalizing a sentence was smaller. This also meant that the complexity of the task was much lower in Arabic than in other languages, and therefore the results for the text normalization tasks in Arabic may not accurately reflect the true text manipulation and processing capabilities of the models tested.

5.2.3.3 Surfacing the strengths of GPT-5

While the performance of GPT-5 on the conversation generation task was underwhelming, we observed very good results on the text normalization task. It managed to score the highest in 6 out of 8 languages, and except for Kinyarwanda, consistently scored 4.5 or higher. On most languages, GPT-5 had less than 1% of low quality (rating 1 or 2) normalized sentences, making it highly suitable for this task.

Given GPT-5's result in this task and the translation task (see Section 5.2.4: Translation), we observe a clear trend of GPT-5 being highly usable across a multitude of languages for text processing and manipulation, but less suitable for content generation tasks. Its performance in long tail languages does not match Gemini 2.5 Pro's performance, but it does show signs of improvement compared to older model generations (such as GPT 4o).

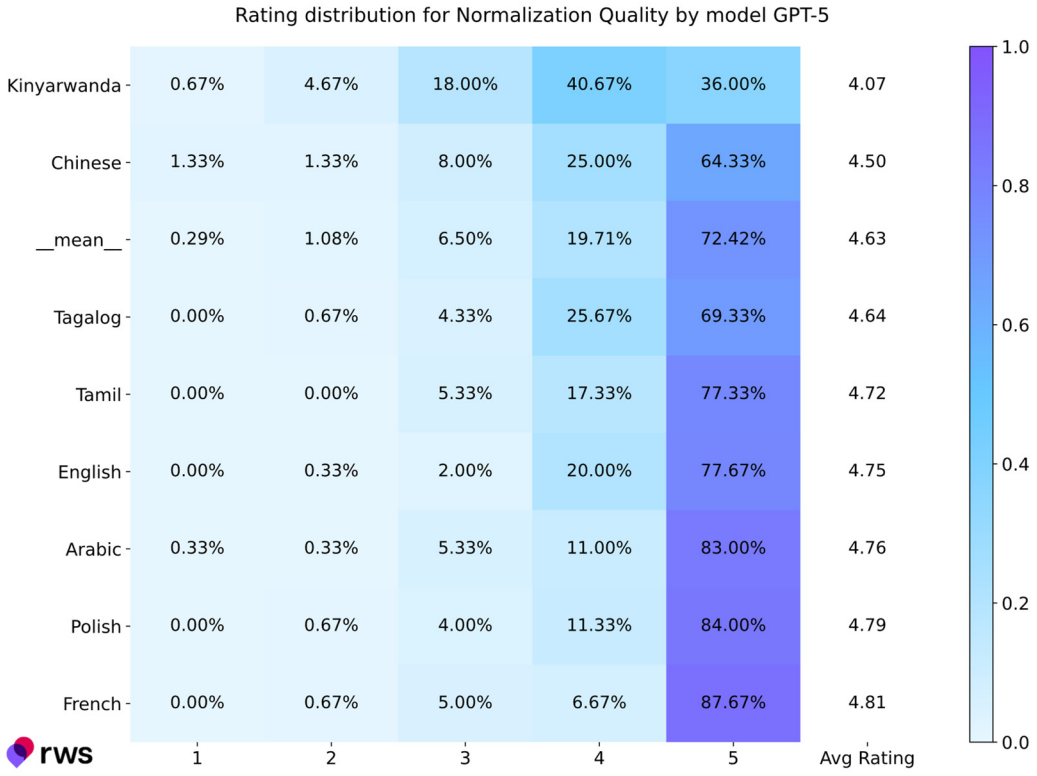

Figure 5-14. Quality rating distribution by language: GPT-5 (text normalization)

5.2.3.4 Gemini 2.5 Pro capable of text normalization in Kinyarwanda

Like the conversation generation task, Gemini 2.5 Pro was the only model that scored well on Kinyarwanda. It scored 4.34 (a 0.27-point lead over GPT-5) with less than 3% of its normalized sentences rated as low quality (rating 1 or 2). While the scores suggest that some level of validation and editing is likely necessary in most production use cases, it signals a shift in LLMs' multilingual capabilities on the long tail end of the language spectrum.

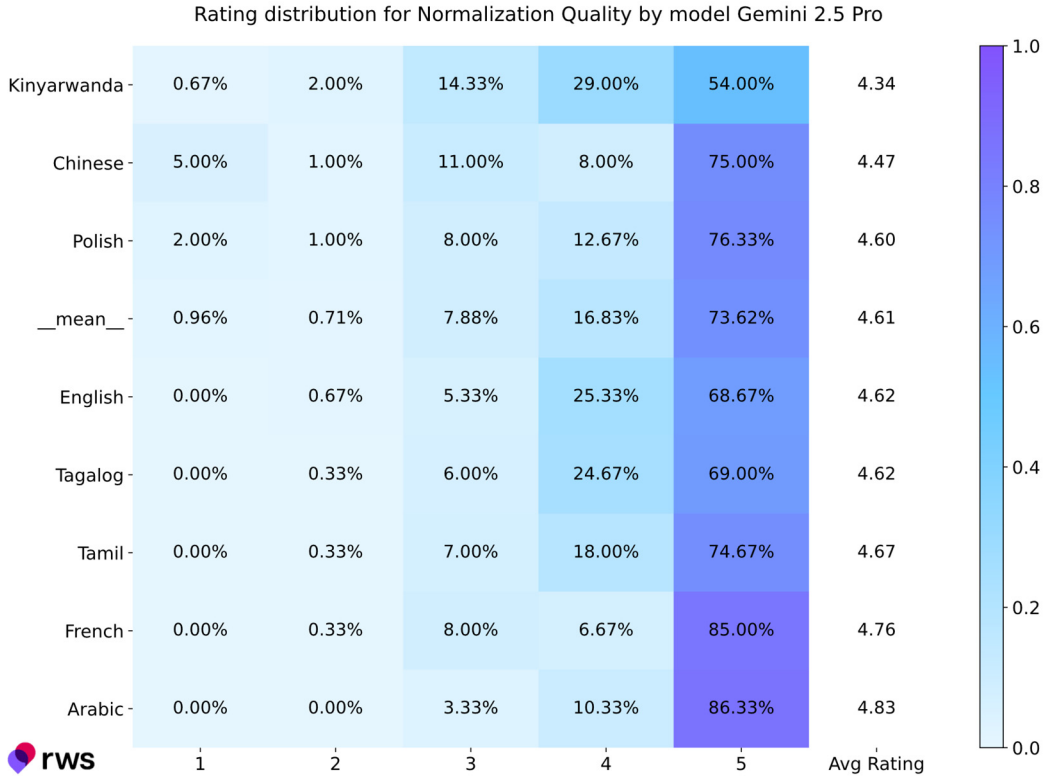

Figure 5-15. Normalization quality rating distribution by language: Gemini 2.5 Pro (text normalization)

5.2.3.5 Model recommendations: text normalization

GPT-5 and Gemini 2.5 Pro both performed well on the text normalization task and are recommended as the best solutions for this type of task. On average, both models scored above 4.5, indicating very strong capabilities for text normalization. GPT-5 scored measurably lower than Gemini 2.5 Pro in Kinyarwanda, suggesting that if long tail language performance is a priority, Google's model is the safer choice.

For text normalization use cases where wide multilingual support is not required, Claude Sonnet 4.5 can also be a good option. The other models scored inconsistently across languages and their average scores dropped well below the 4.5 threshold, making them unsuitable for this type of task.

5.2.4 Translation

Table 5-5. Top performers: translation

Rank	Model name	Average overall score out of 5.0	Minimum overall score out of 5.0	Maximum overall score out of 5.0
1	Gemini 2.5 Pro	4.73	4.48 (zh)	4.84 (fr)
2	GPT-5	4.59	4.39 (zh)	4.73 (fr, ar)
3	Claude Sonnet 4.5	4.47	3.83 (rw)	4.76 (fr)

5.2.4.1 General observations

Translation is a complex task requiring deep knowledge of both the source (English) and the target language. While LLMs are natively multilingual and are increasingly being leveraged in localization workflows, our previous study found they were only usable in a subset of languages. Performance in complex languages such as Kinyarwanda, Tamil, or Chinese, was previously very low.

The current generation of models demonstrates significant improvements in translation capabilities, with most of the models tested delivering very high scores across all languages. This improvement was seen despite the increased complexity of the translation task compared to our previous study (translating paragraphs rather than single sentences).

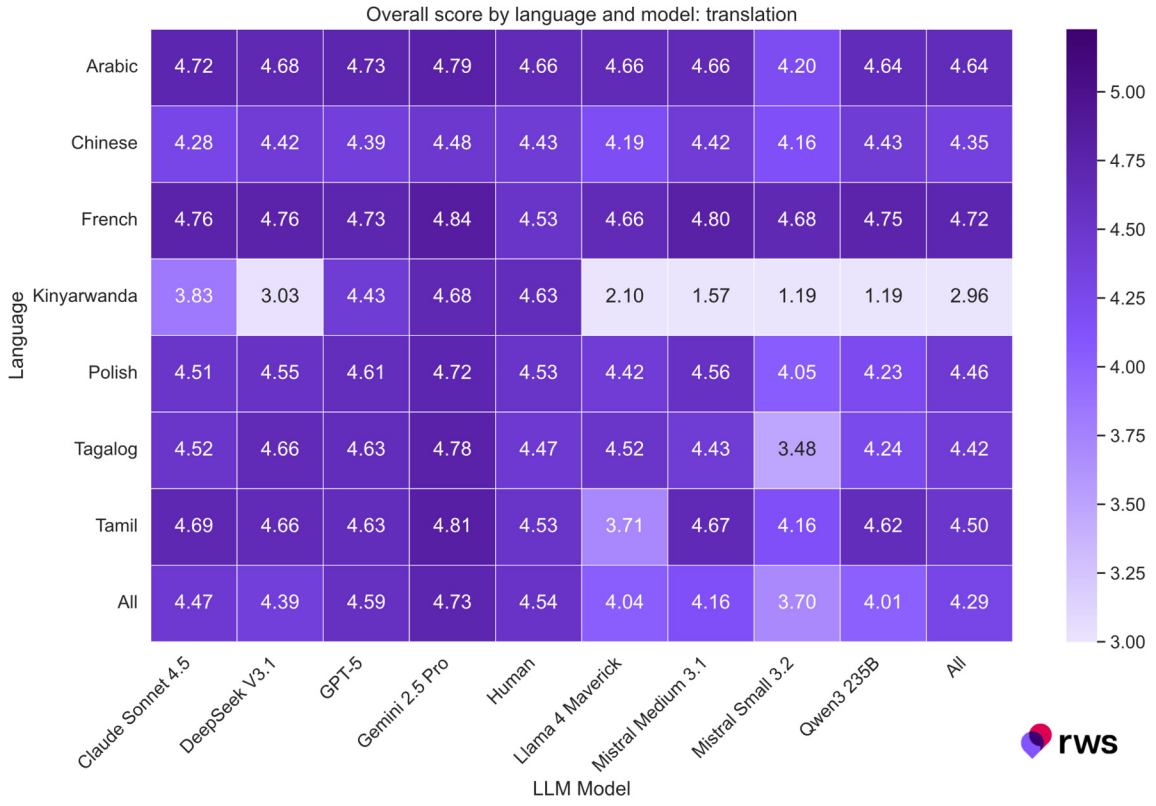

Figure 5-16. Overall score by language and model: translation

The scores were clearly tiered, with differences exceeding 0.1 points among the top 3 ranking models. Notably, the gap between the best and worst performing models was the widest across all metrics tested: more than 1 full point. This stratification highlights the distinct differences between individual model capabilities.

We used three metrics to evaluate outputs: grammar, naturalness, and translation accuracy. The models generally scored the highest on grammar (0.2–0.3 points above the rest). When comparing naturalness and translation accuracy scores, the differences were negligible across most languages, however, for French and Tagalog, naturalness scores were significantly lower (0.29 and 0.20 points respectively) compared to translation accuracy.

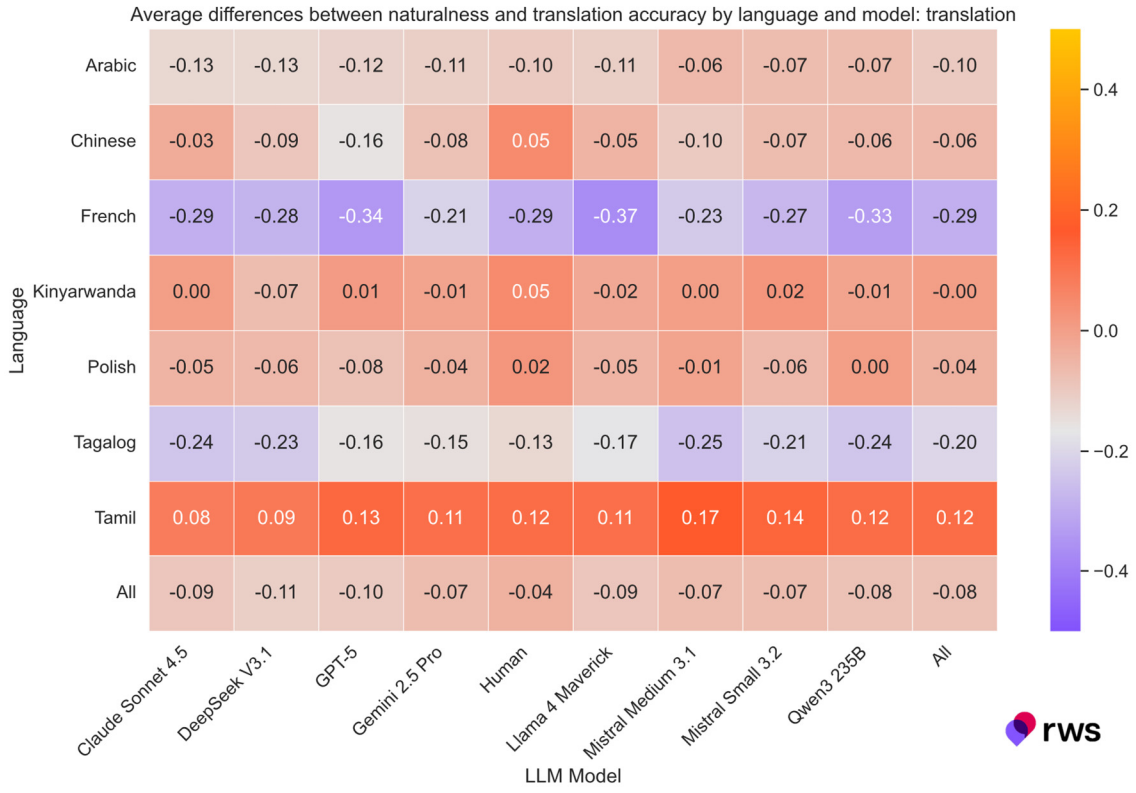

Figure 5-17. Average difference between naturalness and accuracy score by language and model: translation

Finding from our previous study:

One notable aspect of the French ratings was the linguists' sensitivity to naturalness issues. In French, the naturalness score was significantly lower than the grammar score (by 0.2 on average), but this was also the case for the human-sourced content, making it unlikely to be a hallmark of LLM-generated content.

Even with different linguists evaluating, we were able to closely replicate this effect, even to a greater degree: the average difference between grammar and naturalness on the translation task was 0.41. We observed a similar effect on Tagalog. We do not draw any conclusions regarding LLM performance from these findings, but it remains a factor in the analysis that we want to continue to monitor long-term, as it may by symptomatic of quickly evolving registers and linguistic styles.

5.2.4.2 Gemini 2.5 Pro is a strong translation model

Gemini 2.5 Pro emerged as the winner on the translation task, scoring an impressive 4.73 points overall, maintaining a score of 4.5 or higher in 6 out of 7 languages. Its score on the remaining language (Chinese) was 4.48, which is within the margin of error (±0.046).

Regarding its translation accuracy, the proportion of low-quality translations (rating 1 or 2) was less than 2% across all languages. Furthermore, more than half of the languages translated by Gemini 2.5 Pro achieved 95%+ high-quality content (rating 4 or 5).

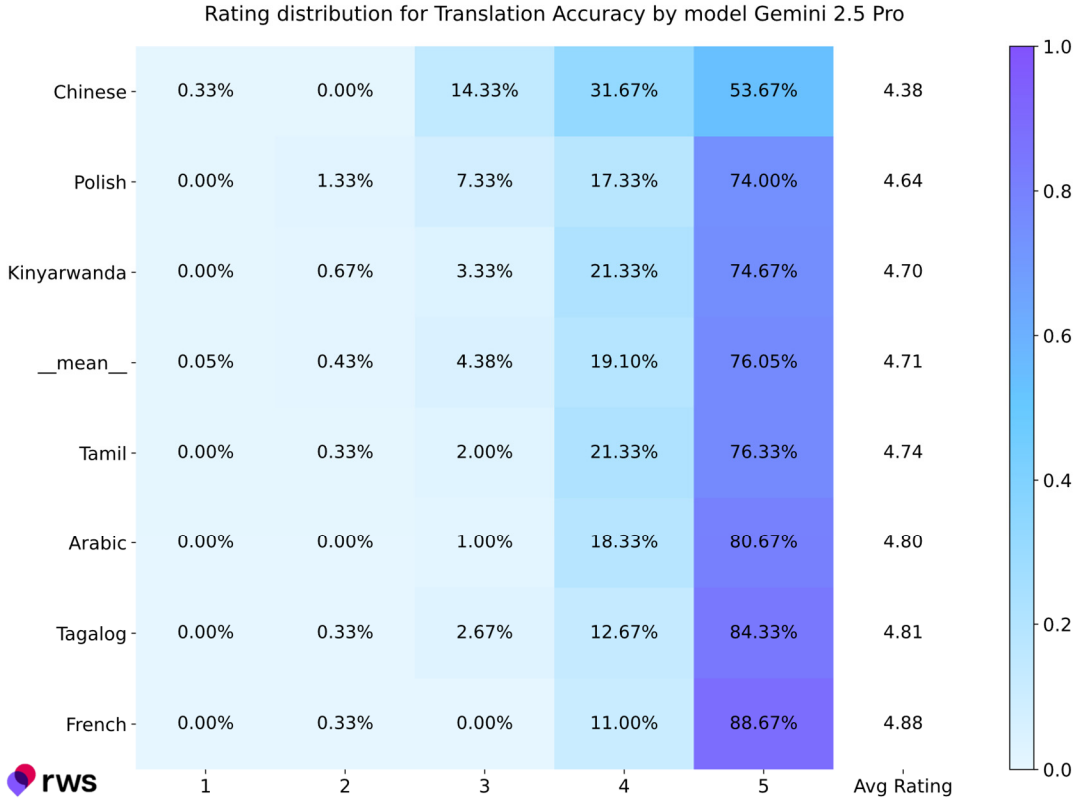

Figure 5-18. Translation accuracy rating distribution by language: Gemini 2.5 Pro (translation)

These results paint a clear picture: Gemini 2.5 Pro is a very strong model for translation, even extending its capabilities to long-tail and underrepresented languages. Given Google's heritage in translation and the promising results of Gemini 1.5 Pro, this is not entirely surprising; however, the velocity at which support for long-tail languages is improving is notable.

5.2.4.3 Translation tasks require large models

As mentioned previously, translation is a complex task requiring extensive knowledge of vocabulary, morphology, cultural references, and connotations across languages. There is a critical amount of raw knowledge that any system needs to function effectively in a language. Our findings suggest that this "critical mass" does not transfer well when compressed via lossy methods – such as model distillation or, more generally, by representing linguistic patterns within the constrained parameter space of a smaller neural network.

It's important to note that, while specialized LLMs focused solely on translation can provide high quality outputs even at lower parameter counts, our discussion here focuses strictly on general-purpose models.

Given these principles, we were not surprised to observe a sharp decrease in scores associated with smaller models. Although exact parameter counts for many closed models are not public, it is speculated that the largest state-of-the-art models exceed 1 trillion parameters. The following table summarizes the relationship we observed between model size and translation performance. Note that we are not providing an estimate of Mistral Medium 3.1's size, but its performance is likely above its weight class.

Table 5-6. Relationship between model size and performance: translation

Rank	Model name	Score (translation accuracy)	Model size
1	Gemini 2.5 Pro	4.71	Estimated at 1T or more
2	GPT-5	4.58	
3	Claude Sonnet 4.5	4.45	
4	DeepSeek V3.1	4.37	671B
5	Mistral Medium 3.1	4.12	N/A
6	Llama 4 Maverick	3.99	400B
7	Qwen3 235B	3.98	235B
8	Mistral Small 3.2	3.66	24B

Notably, bigger isn't always better for all tasks. For instance, we see almost identical rankings for the text normalization task (which arguably requires deep linguistic knowledge), whereas the ranking on data generation tasks like domain-specific paragraph generation or conversation generation is more varied, with smaller differences between models.

However, for translation, the trend is clear. Even though modern LLMs utilize increasingly complex architectures and post-training procedures, our findings confirm that translation capabilities and similar complex language manipulation skills are directly correlated with model size for general-purpose LLMs. While training small models for specific languages (especially long-tail ones) remains a viable option, these efforts will face increasing pressure from general-purpose commercial SOTA models, which are rapidly achieving high-level support for a vast scope of languages.

5.2.4.4 High floor, low ceiling on Chinese

The average rating across all models for translation accuracy by task in Simplified Chinese was 2nd from the lowest, even though the volume of data available in Chinese is similar to languages like French. While the volume of training data in a language is not the only indicator of its performance in text generation and manipulation benchmarks, it is usually closely correlated. In our previous study, Chinese ranked 4th on the translation task, trailing Arabic in 3rd place.

This time, performance was relatively strong (all models scored above 4.0), but with low differentiation: the delta between the worst and best scoring models was only 0.37 points.

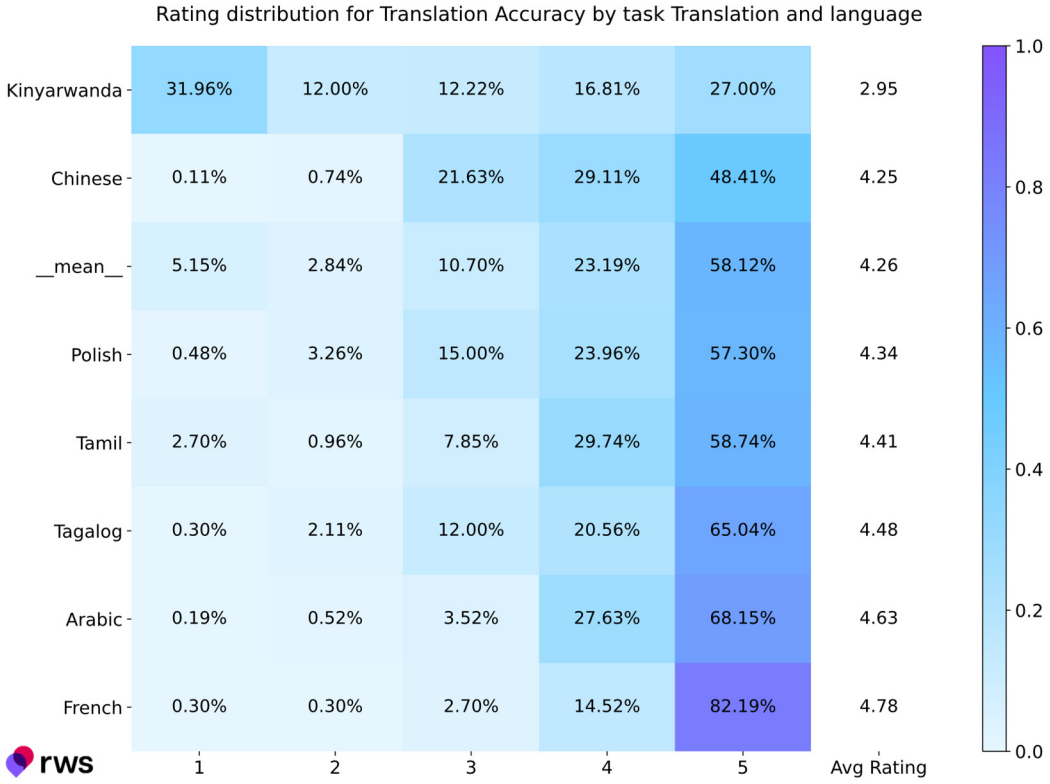

Figure 5-19. Accuracy rating distribution by language: translation

We analyzed the data for possible causes but found no clear evidence. For other tasks, our annotators ranked Simplified Chinese LLM outputs both very low (see Section 5.2.2.3: GPT-5's underwhelming performance as an example) and very high (the best-performing models scored as high as 4.82 on other tasks). Common-sense expectations held true: Chinese models (Qwen 3 and DeepSeek) ranked well (2nd and 3rd), and the smallest model (Mistral Small 3.2) ranked last.

Regarding annotator agreement, we observed an anomaly: annotators were most aligned on naturalness and least aligned on grammar – the opposite of other languages. We also observed distinct cohorts in the translation accuracy ratings, with one group consistently rating higher (4s and 5s) than the other (3s and 4s). The prevalence of low scores (1 and 2) was very low in both cohorts – less than 2%.

5.2.4.5 Model recommendations: translation

Our recommendation for translation is Gemini 2.5 Pro, which scored 4.73 points overall and provided high-quality translations even in Kinyarwanda – a feat that was unattainable by older generations of the models.

GPT-5 is also a reliable option, scoring well (4.43) even in Kinyarwanda and 4.58 overall across all languages. For a smaller subset of well-supported languages, Claude Sonnet 4.5, DeepSeek V3.1 and Mistral Medium 3.1 are viable, though they incur a quality penalty compared to Gemini 2.5 Pro.

5.3 Performance by language

5.3.1 Supported vs. unsupported languages

As previously noted, our language selection strategy was designed to sample a broad spectrum of linguistic complexity, allowing for performance extrapolation to similar languages. We deliberately disregarded the specific support lists of individual foundational LLMs during this selection process to ensure an unbiased evaluation.

Determining the official language support for a given LLM remains a complex challenge. Many AI labs do not publish comprehensive lists, often relying on vague descriptors such as "supports 100+ languages" rather than specific enumerated capabilities. We consider this opacity actively detrimental to the adoption of LLMs within diverse linguistic communities and strongly urge AI labs to commit to publishing transparent lists of officially supported languages. In this regard, we commend Google and Alibaba for their transparent approach in providing specific documentation.

Beyond official support, LLMs often acquire varying degrees of fluency in a wider array of languages, likely a byproduct of multilingual inclusion in massive training datasets like The Pile or Common Crawl. Consequently, the following overview represents our best estimation of language support across the tested models. Please note that due to the transparency issues highlighted above, this overview relies on informed assessments; instances where we hold lower confidence are indicated with a question mark.

Table 5-7. Estimated language support by model

Model name	Arabic	Chinese	English	French	Kinyarwanda	Polish	Tagalog	Tamil
Claude 4.5 Sonnet	✓	✓	✓	✓	?	✓	✓	✓
DeepSeek V3.1	✓	✓	✓	✓	✗	✓	✓	✓
Gemini 2.5 Pro	✓	✓	✓	✓	✗	✓	✓	✓
GPT-5	✓	✓	✓	✓	✗	✓	✓	?
Llama 4 Maverick	✓	✓	✓	✓	✗	✗	✓	✗
Mistral Medium 3.1	✓	✓	✓	✓	✗	✓	?	?
Mistral Small 3.2	✓	✓	✓	✓	✗	✓	?	?
Qwen 3 235B	✓	✓	✓	✓	✗	✓	✓	✓

In this iteration of the study, we found no egregious instances of LLMs claiming support for a language while failing to deliver usable results (overall scores < 4.0).

Conversely, we observed several examples where models demonstrated robust proficiency in languages they do not officially support. Notably, Llama 4 Maverick performed strongly in both Polish (4.31) and Tamil (4.33). Similarly, Gemini 2.5 Pro achieved an impressive 4.56 in Kinyarwanda, while GPT-5 delivered a respectable 4.21 in the same language, despite it likely being unsupported.

In contrast, Claude Sonnet 4.5 scored below 4.0 in Kinyarwanda; however, due to the vagueness of its "200+ supported languages" designation, it remains unclear whether this represents a performance gap in a supported language or simply the natural limit of an unsupported one.

5.3.2 Model performance: Arabic

We observed generally very high performance in Arabic across all tasks. The new generation of models can generate and manipulate Arabic text with high proficiency, marking a significant improvement over previous versions. This trend holds true even for smaller models; for instance, Mistral Small 3.2 achieved a slightly lower but still robust overall score of 4.45.

The three top-performing models were Gemini 2.5 Pro, Claude Sonnet 4.5, and DeepSeek V3.1. Notably, the field is extremely competitive: the score spread between the 1st and 7th place was only 0.17 points.

However, a significant disparity emerges when looking at operational efficiency. While Claude Sonnet 4.5 rivals the best models in quality, it exhibits poor tokenizer efficiency in Arabic, encoding only 1.48 characters per token. In contrast, DeepSeek V3.1 (2.63 chars/token) and Gemini 2.5 Pro (3.23 chars/token) are significantly more efficient.

This means that for high-volume Arabic workloads, using Claude Sonnet 4.5 could be more than twice as expensive as Gemini 2.5 Pro for generating the exact same amount of text, purely due to token consumption.

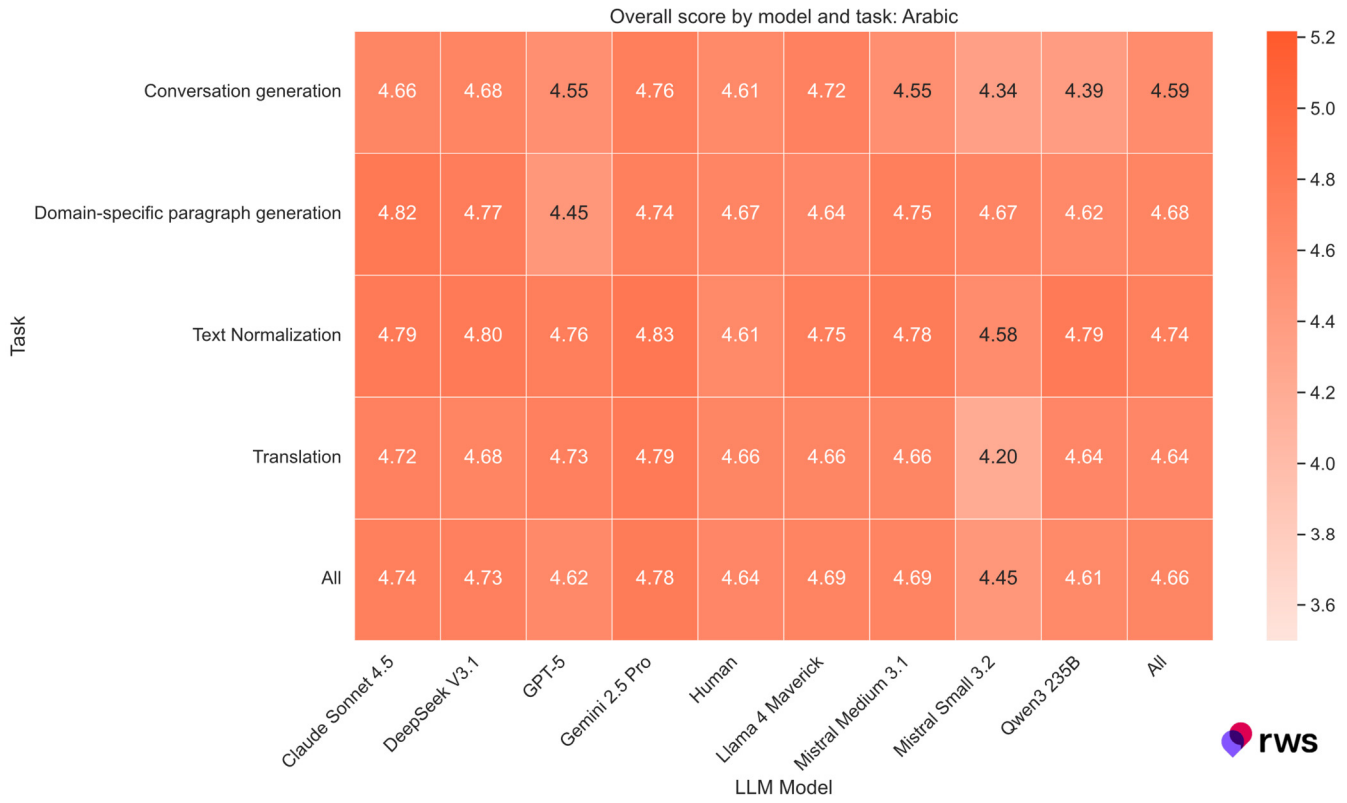

Figure 5-20. Overall score by model and task: Arabic

5.3.3 Model performance: Chinese

Simplified Chinese exhibited one of the most dramatic disparities between content generation and text manipulation capabilities. With the notable exception of GPT-5, all models were highly capable of producing high-quality text in Chinese, including Mistral Small 3.2, which scored very highly in both data creation tasks.

However, the text normalization and translation tasks proved to be significant bottlenecks for most models. Only GPT-5 and Gemini 2.5 Pro scored consistently high (close to 4.5 points), while smaller models performed very poorly on these complex tasks.

Tokenizer efficiency is also a critical factor. Claude Sonnet 4.5, despite scoring generally well in Chinese, suffers from inefficient tokenization: it encodes on average only 0.85 characters per token, compared to Gemini 2.5 Pro's 1.59. This nearly doubles the token consumption for the same amount of text.

Considering the performance gap in manipulation tasks and the efficiency issues of competitors, the only model we can recommend for general-purpose use in Chinese is Gemini 2.5 Pro.

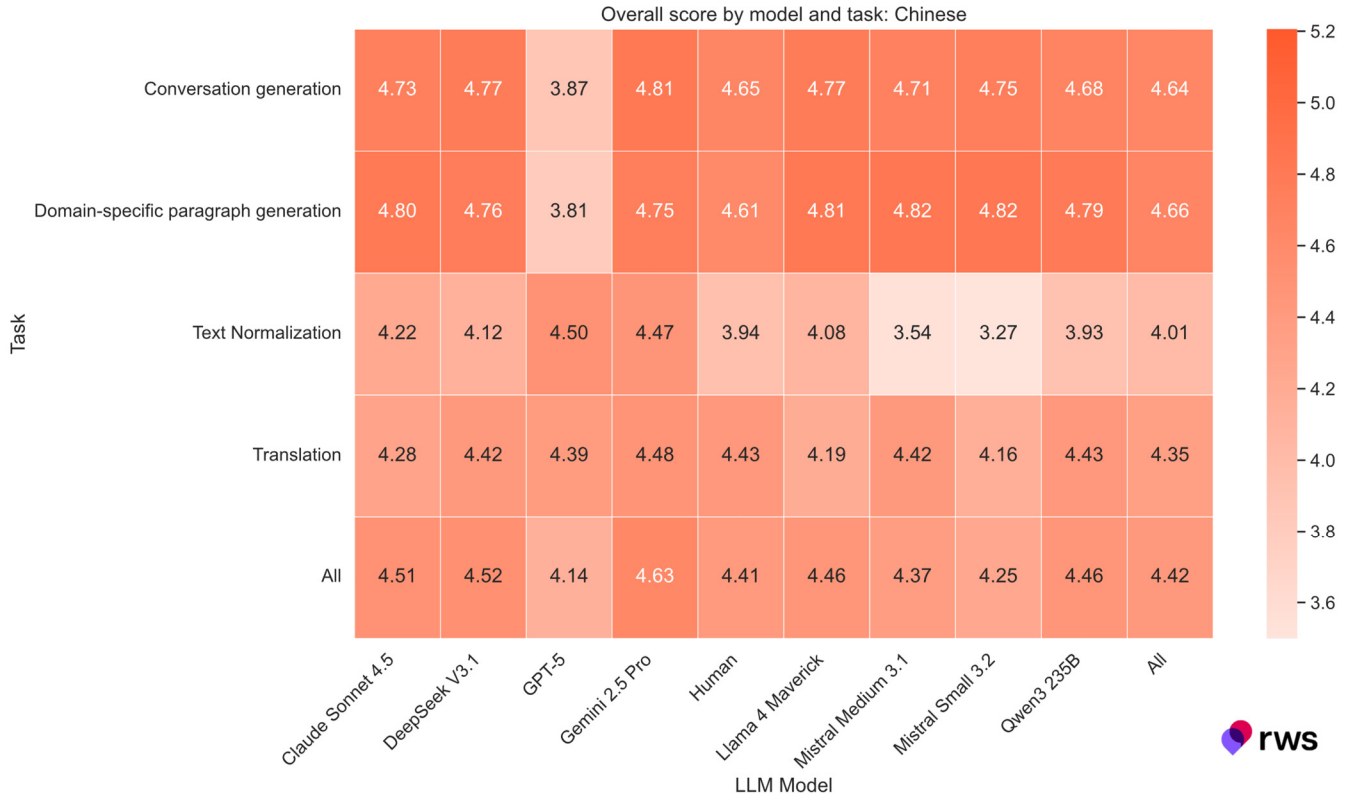

Figure 5-21. Overall score by model and task: Chinese

5.3.4 Model performance: English

Traditionally, models perform best in English, and it therefore serves as a good baseline for comparing performance across other languages. Consequently, it is unsurprising that output quality for content creation tasks in English was nearly perfect across all models – a finding consistent with our previous study. Modern models are heavily trained on English corpora and demonstrate exceptional capability in producing high-quality English outputs.

However, text normalization scores displayed higher variance, with a performance gap of nearly 0.5 points between the first and last models. Surprisingly, this was one of the few instances where Gemini 2.5 Pro only just narrowly secured a top 3 position (tying with DeepSeek V3.1).

Tokenizer efficiency in English was consistently high across all models, with only negligible differences. For English tasks, we recommend Claude Sonnet 4.5, DeepSeek V3.1, and Gemini 2.5 Pro

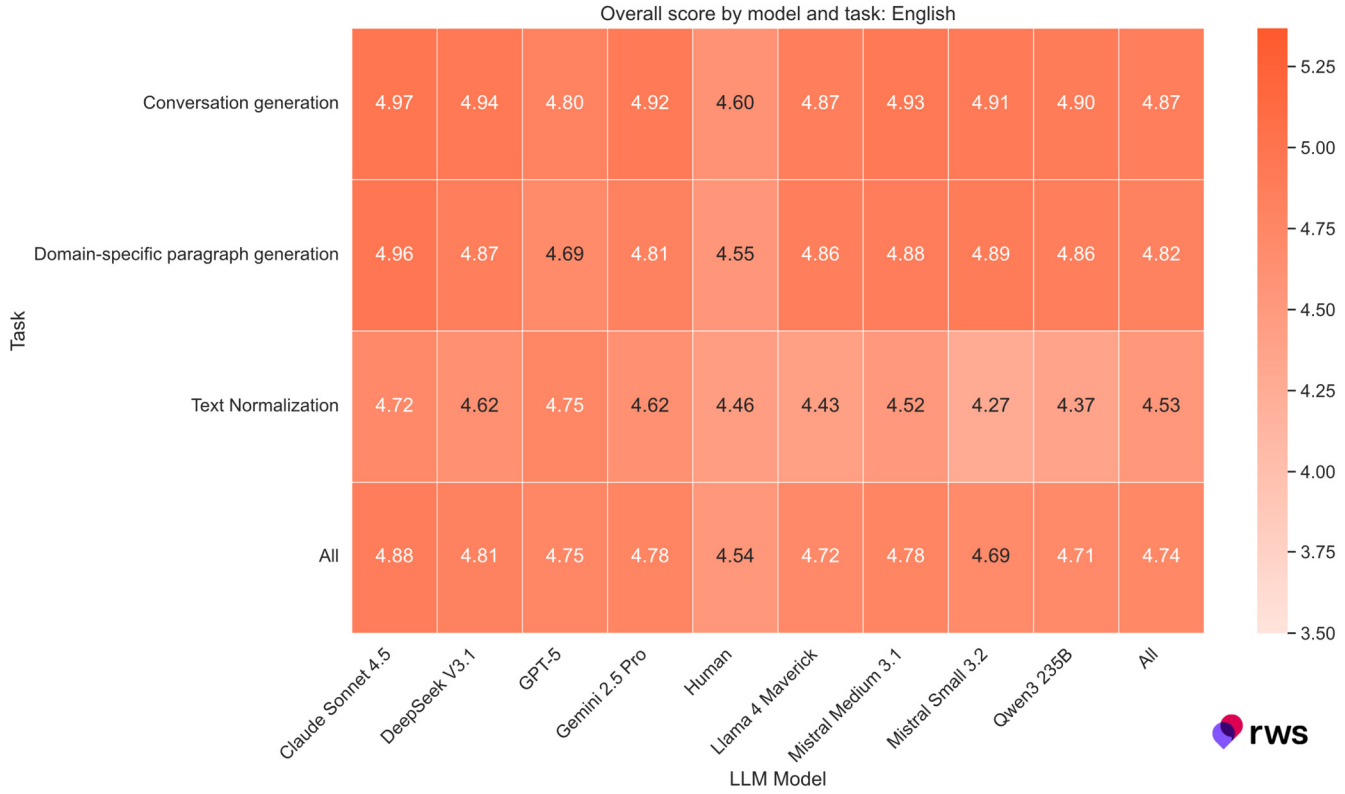

Figure 5-22. Overall score by model and task: English

5.3.5 Model performance: French

French scores typically track closely with English results, and this pattern was confirmed in our study. Except for the text normalization task, models scored very highly (above 4.5 points overall) across the board in French.

As a French-native model, Mistral was expected to perform exceptionally well. While both Mistral models scored highly on content generation and translation tasks, their overall standing was impacted by slightly weaker performance on the text normalization task compared to top-tier global models.

One of the differentiators in French is tokenizer efficiency. Claude Sonnet 4.5 showed relatively poor efficiency, encoding only 3.33 characters per token compared to Gemini 2.5 Pro's 4.80. This has direct economic implications: generating the same volume of French text with Claude Sonnet 4.5 requires roughly 45% more tokens than with Gemini 2.5 Pro, significantly increasing inference costs for large-scale operations.

Based on these findings, our primary recommendations for French models are Gemini 2.5 Pro, which offers the best balance of efficiency and quality, followed closely by Claude Sonnet 4.5 and GPT-5. For simpler use cases where the highest tier of performance is not strictly necessary, DeepSeek V3.1 and Mistral Medium 3.1 also serve as strong, cost-effective alternatives.

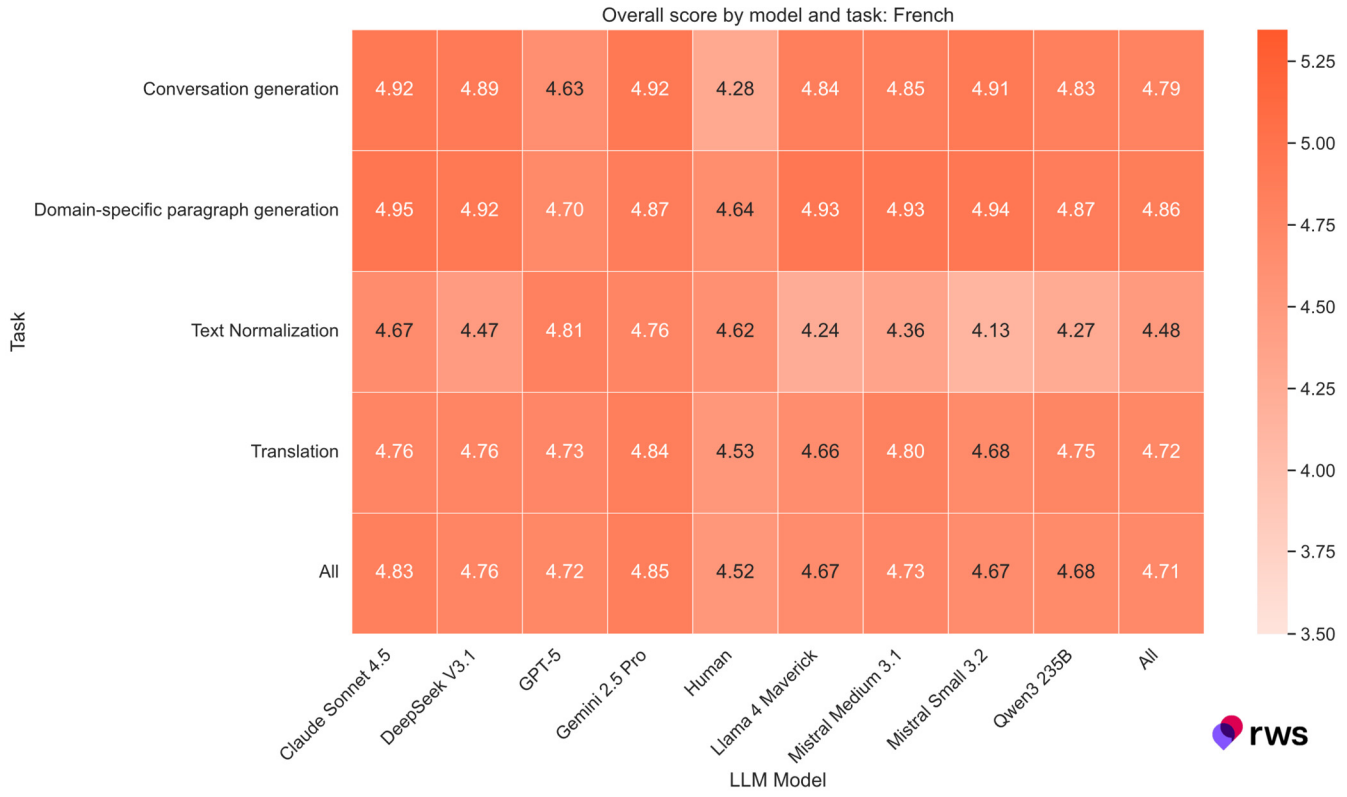

Figure 5-23. Overall score by model and task: French

5.3.6 Model performance: Kinyarwanda

Kinyarwanda serves as our representative for long-tail languages, acting as a litmus test for LLM capabilities in underrepresented linguistic regions. In our previous study, scores were universally low, leading to a recommendation against using LLMs for this language cohort. However, newer model generations have fundamentally altered this landscape.

We observed respectable performance in Kinyarwanda from GPT-5, particularly in domain-specific paragraph generation and translation. Yet, the most significant finding was the performance of Gemini 2.5 Pro, which achieved an overall score of 4.56. This result demonstrates a high level of proficiency in Kinyarwanda; notably, it even slightly outperformed our single-pass human creation baseline.

Consequently, the only model we can unreservedly recommend for Kinyarwanda (and potentially other long-tail languages) is Gemini 2.5 Pro. That said, results from models like GPT-5 and, to a lesser extent, Claude Sonnet 4.5, indicate a broader trend: the newest generation of frontier models is rapidly gaining fluency across the long tail of the linguistic spectrum.

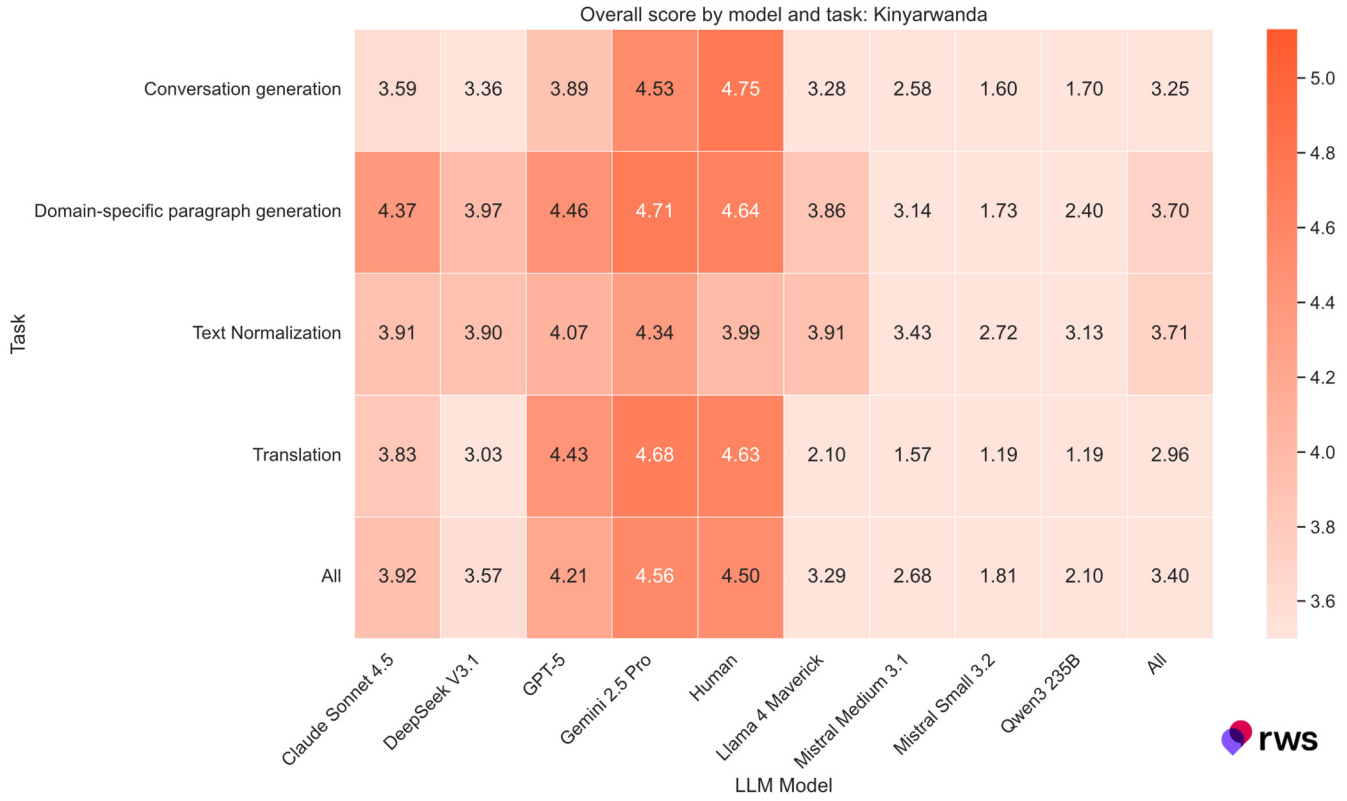

Figure 5-24. Overall score by model and task: Kinyarwanda

5.3.7 Model performance: Polish

Polish presented some of the study's most unexpected results. In conversation generation specifically, several models scored below 4.0, suggesting that creating natural, grammatically sound dialogue is a significant challenge in Polish. Although this volatility was less pronounced in domain-specific paragraph generation, we strongly advise against using models that scored below 4.0 for synthetic data generation or content creation in Polish.

Performance on text manipulation tasks was generally more balanced, yet distinct weaknesses emerged. Mistral Small 3.2 struggled with text normalization (3.96), followed closely by Llama 4 Maverick and Qwen3 235B (tied at 4.14). We also observed an interesting trade-off between task types among top models: Claude Sonnet 4.5, despite ranking 1st in content generation, lagged slightly behind the leaders in text manipulation tasks, though it still maintained robust scores above 4.5. In a complete reversal of this pattern, GPT-5 performed poorly on content generation, but secured 1st and 2nd place in text manipulation tasks.

Regarding tokenizer efficiency, we did not observe any meaningful differences across the tested models in Polish.

Ultimately, the only models to demonstrate consistent excellence with scores above 4.5 across the board were Gemini 2.5 Pro and Claude Sonnet 4.5. These represent our primary recommendations for Polish, regardless of task complexity. For scenarios requiring a balance between performance and budget, DeepSeek V3.1 and Mistral Medium also deliver strong results and serve as viable, cost-effective alternatives.

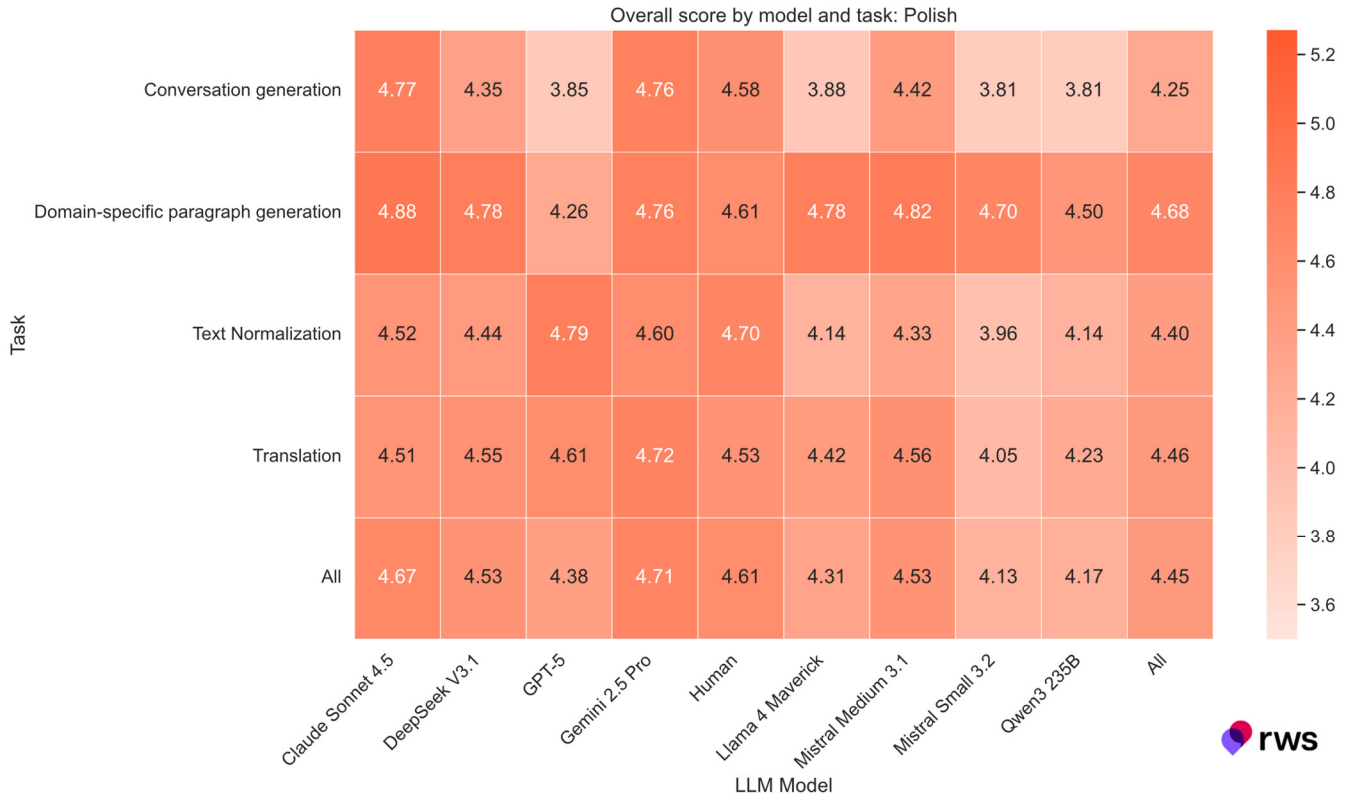

Figure 5-25. Overall score by model and task: Polish

5.3.8 Model performance: Tagalog

Tagalog marks another decisive victory for Gemini 2.5 Pro. However, unlike several other languages, GPT-5 also demonstrated robust performance across the board, joining Gemini Pro in an exclusive tier of models scoring above 4.5 on every task in Tagalog. Claude Sonnet 4.5 narrowly missed this threshold, scoring 4.46 in text normalization; however, its exceptional performance in content creation tasks pushed its overall aggregate score higher than that of GPT-5.

Other models also showed strong potential. DeepSeek V3.1, Mistral Medium 3.1, and Llama 4 Maverick all achieved overall scores above 4.5 in Tagalog, with only minor struggles in the text normalization task. Conversely, Qwen3 235B displayed unbalanced performance, scoring well in most categories but dropping significantly below 4.0 in text normalization. Mistral Small 3.2 proved effectively unusable for text manipulation in Tagalog, although its content creation capabilities remained potentially viable.

Regarding tokenizer efficiency, we did not observe any meaningful differences across the tested models on Tagalog.

Our top recommendation for Tagalog is Gemini 2.5 Pro for its consistent excellence. However, the field is competitive: Claude Sonnet 4.5, DeepSeek V3.1, and GPT-5 also performed impressively and stand as strong alternatives for Tagalog depending on specific task requirements.

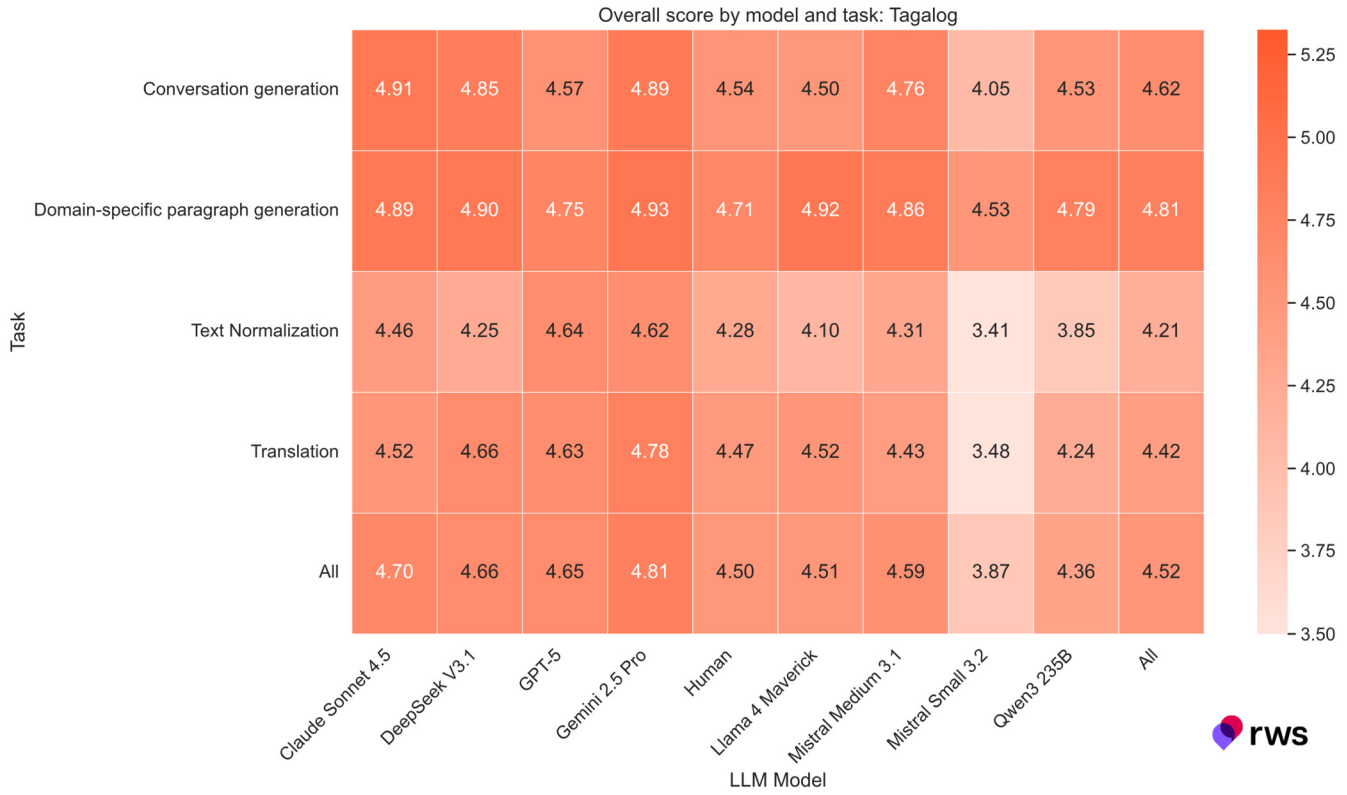

Figure 5-26. Overall score by model and task: Tagalog

5.3.9 Model performance: Tamil

Tamil presents unique challenges for LLMs due to its complex script and linguistic structure; in our previous study, it ranked alongside Kinyarwanda as one of the most difficult languages. If our results for Kinyarwanda in this study suggests that SOTA models are gaining proficiency in long-tail languages, our Tamil findings confirm this trend and extend it well beyond the very top tier of models. Four models scored above 4.5 points overall, indicating surprisingly broad support. Moreover, we observed further confirmation of this trend as multiple models surpassed the quality of our single-pass human baseline.

These high scores do not stem from an overly optimistic annotator pool: in 3 out of 4 tasks, at least one model scored below 4.0. The roster's smallest model, Mistral Small 3.2, scored just above 4.0, and we generally advise against its use for Tamil. Larger models also had specific stumbling blocks: Llama 4 Maverick and GPT-5 achieved high scores but scored below 4.0 in conversation generation and translation respectively.

However, tokenizer efficiency remains the critical differentiator for Tamil. We observed a staggering 4x difference in character-per-token efficiency between the worst performer (Qwen3 235B at 0.89) and the best (Gemini 2.5 Pro at 4.24). Claude Sonnet 4.5 also performed poorly at 1.19. This inefficiency makes it difficult to recommend Claude (or Qwen) for any substantial Tamil workloads, as the cost implications are severe compared to Gemini.

Our primary recommendation is Gemini 2.5 Pro, which achieved a remarkable overall average of 4.72 and consistently scored above 4.5. For budget-conscious use cases, DeepSeek V3.1 and Mistral Medium 3.1 serve as strong alternatives.

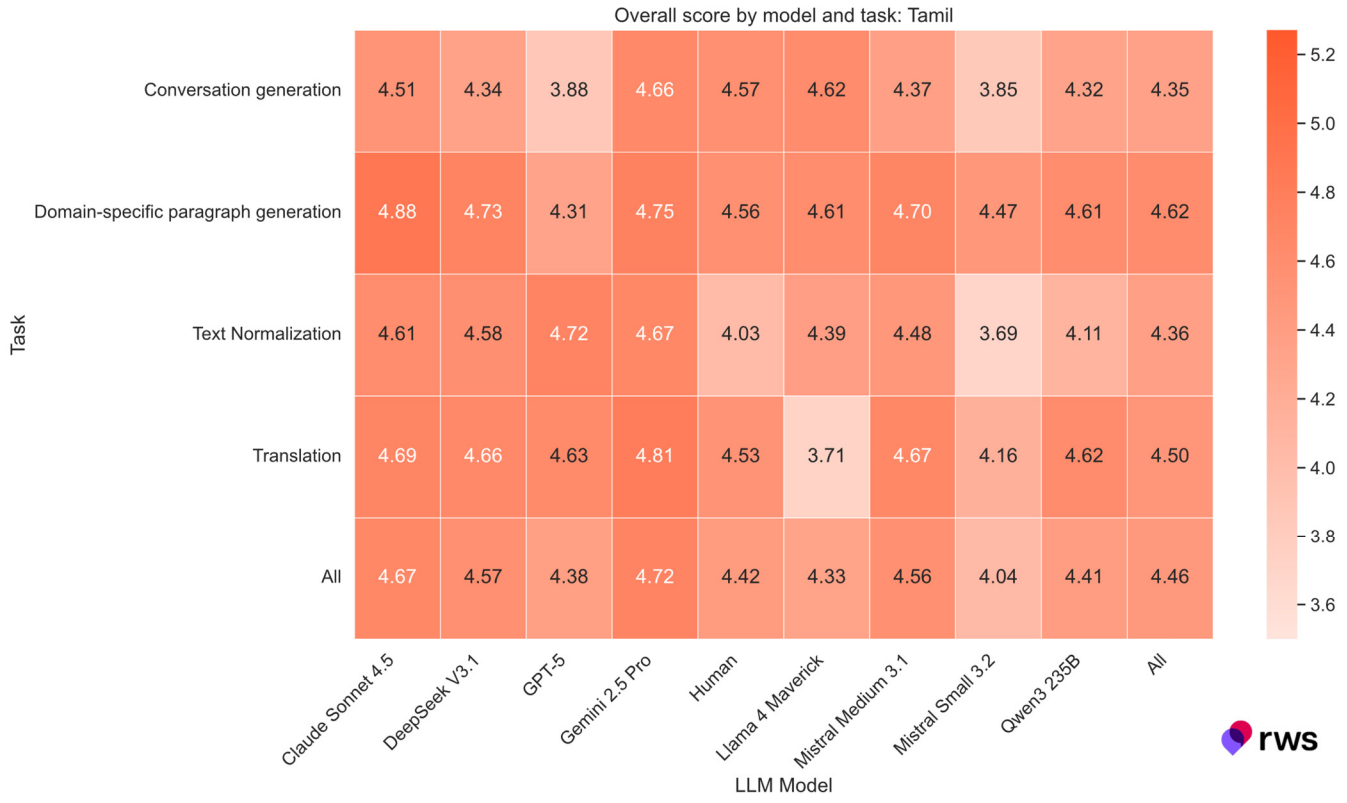

Figure 5-27. Overall score by model and task: Tamil

5.4 Annotator agreement

5.4.1 Challenges with human agreement

Human annotation has clear advantages in providing real preferences, as opposed to proxy preferences from an LLM judge. However, it also comes with inherent challenges: as the number of annotators increases, so does the potential for disagreement stemming from small differences in calibration or divergent world views.

Languages also have varying degrees of standardization, numerous dialects and regional variants, and influences from other languages. Moreover, sociolinguistic and sociopolitical aspects influence ratings: what might seem a perfectly natural way of expressing oneself to one generation of annotators could appear unnatural to another. All these factors play a significant role, especially when assessing the grammatical correctness or naturalness of text.

A standard method for reducing variance in annotator outputs is replication – having multiple annotators rate the same output. In this study, we used a replication factor of 3 and observed annotator agreement scores on individual languages.

5.4.2 Annotator agreement analysis

Given that we used more annotators than in our previous study, we had concerns about the potential impact of the larger pool on annotator agreement levels. We used Pearson's r, Spearman's ρ, and Krippendorff's α correlation metrics to measure inter-annotator agreement. We found that correlation scores were similar to our previous study, with most languages falling into the 'Moderate Correlation' category. This is likely due to the qualification stage, during which we selected annotators who demonstrated a strong understanding of the tasks and the grading rubric (see Section 2.1.2.2: Selecting creators for methodology details applicable to both creators and annotators).

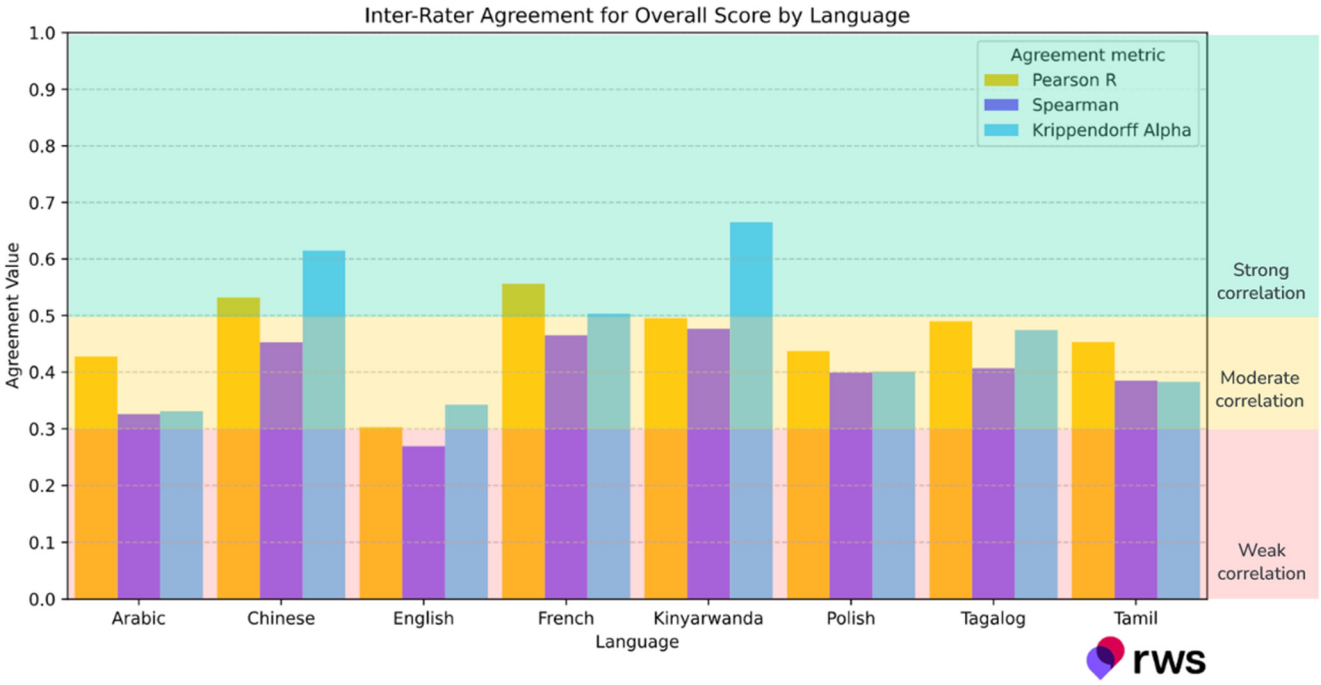

Figure 5-28. Inter-annotator agreement: overall score by language (r, ρ, α)

In our previous study, 2 languages fell into the ‘Weak Correlation’ category: English and Chinese. For Chinese, scores in the last study were affected by rater misalignment at the start of the project; in this study, however, Chinese alignment was among the highest we measured.

Agreement on English followed the same pattern as in the previous study: scores were among the lowest, with naturalness agreement being particularly low. We observed one cohort of raters giving lower scores (a higher incidence of rating 4) than the rest of the pool.

From a methodological standpoint, the issue with English ratings appears to be attributable to a clustering of scores at the maximum rating of 5 (a 'ceiling effect'). Most models perform well on English tasks; consequently, few received ratings below 4, with most ratings being 5. In such circumstances, annotator agreement metrics become less reliable and may not produce statistically relevant results. Moreover, comparing annotator agreement metrics for each LLM separately revealed that human-created content, which had the highest proportion of ratings below 5, had the highest levels of annotator agreement across all metrics.

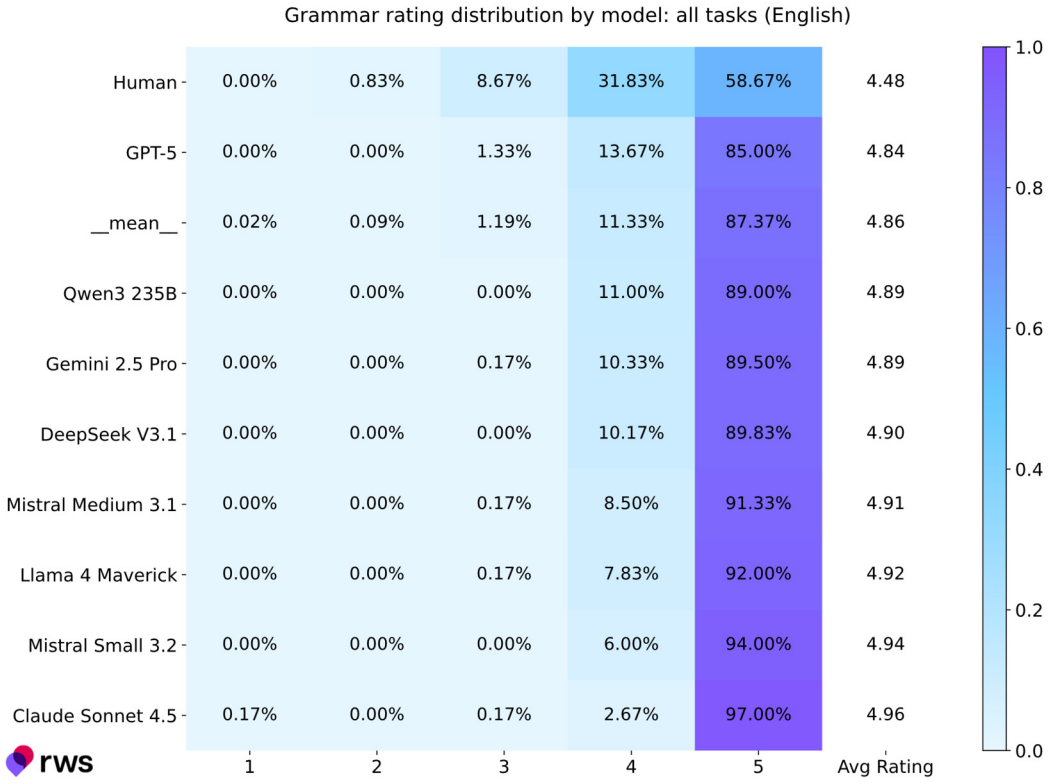

Figure 5-29. Grammar rating distribution by model: all tasks (English)

We also analyzed the score spread by taking the highest and lowest rating for each sentence and calculating the difference. The distribution of the score spread is visualized in the following chart using a violin plot. The width of each curve represents the frequency for the given spread, the black bar represents the interquartile range, and the white stripe marks the median value. Because the overall score is calculated by averaging 3 or 4 individual metrics, some values are not whole numbers.

This metric adds another perspective to annotator agreement since it quantifies the magnitude of divergence. A high spread indicates that when annotator judgments differed, they differed by multiple points on the rating scale. A low spread indicates that in such situations, the ratings were likely just one point apart.

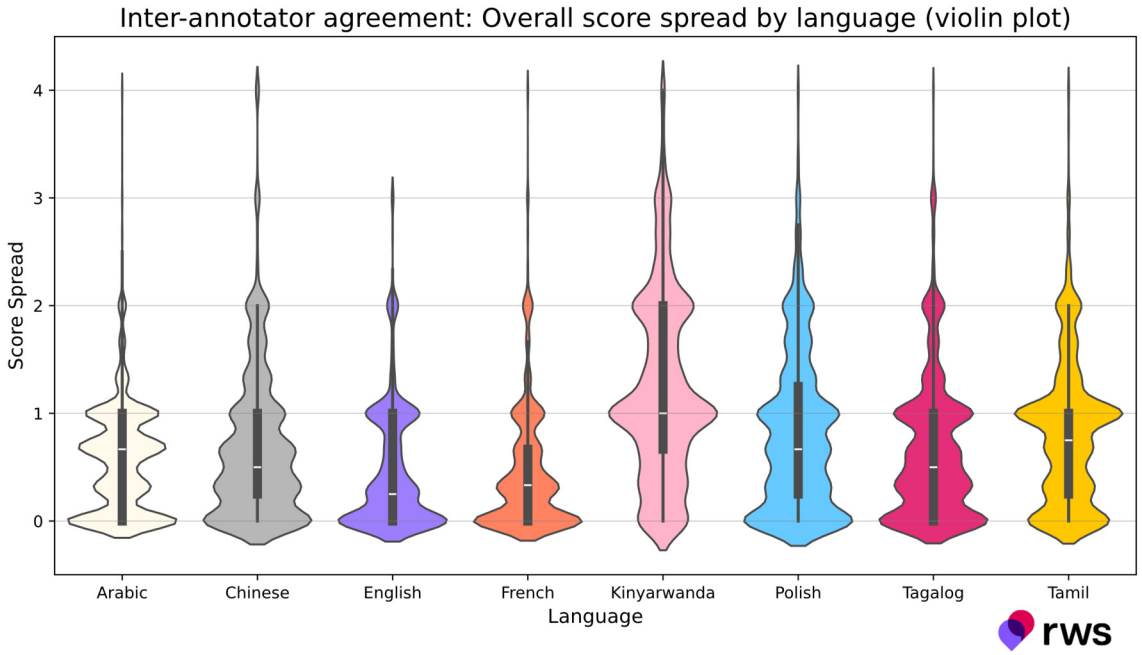

Figure 5-30. Inter-annotator agreement: overall score spread by language (violin plot)

The violin plot suggests that, in most languages, disagreement among the 3 annotators for each LLM output resulted in a score spread between 0 and 1, indicating a healthy consensus. Interestingly, languages such as Chinese or Kinyarwanda, which had high annotator agreement scores, display a longer tail in the score spread compared to other languages. This suggests that while agreement was generally high, the minority of disagreements were more severe (exceeding 1 point), implying annotators disagreed on multiple metrics simultaneously.

Overall, given the complexity of the tasks and the scale of the annotation effort, annotator agreement scores are within our expectations and generally in line with the agreement rates and score spreads observed in our previous study. We believe these results constitute a solid basis for evaluating the quality of LLM-generated content.

5.5 Humans vs. LLMs

In previous sections, we refrained from commenting on scores for text created or processed by humans to centralize the discussion, as we believe this comparison requires a different context compared to a head-to-head LLM evaluation.

First, it is crucial to reiterate that scores marked as 'Human' do not represent the absolute theoretical peak of human performance. Our focus is on general-purpose content creation and manipulation at scale, and assessing the viability of LLMs for these tasks across languages. Our human creators operated under time constraints without a secondary QA phase. In real-world scenarios, rigorous vetting and quality assurance loops would be employed to the extent that a project's scale allows.

Our primary research question was whether specific scenarios exist in data creation and manipulation where LLMs offer a more efficient alternative while maintaining acceptable quality – with the expectation that all content, whether human or LLM-generated, would undergo final validation by a professional.

Second, we intentionally did not measure performance in highly specialized domains (such as high-end marketing copy, transcreation, or technical domain translation), where human professionals provide unique, irreplaceable value. The source texts for translation were of moderate complexity, requiring neither domain expertise nor deep specialized knowledge.

Furthermore, our translation task did not mandate adherence to provided terminology glossaries or legacy translation memory (fuzzy matches). While general-purpose translation represents a significant volume of daily output from both professionals and automated systems, the majority of economic value in the localization industry remains concentrated in specialized workflows that leverage specific tooling and expert knowledge.

5.5.1 Humans vs. LLMs: data creation speed

One major motivation for creating data synthetically is the lack of availability or high cost of acquiring legally compliant data in specific domains. When creating content from scratch using human writers, the time required to generate content at scale is also a major concern.

Naturally, humans cannot match the speed of LLMs in text creation. Human creation times for paragraph-long texts averaged between 240 and 313 seconds, whereas LLMs produced similar outputs within 5–30 seconds. Moreover, LLM requests can be processed in parallel, effectively bringing the creation time per paragraph down to single second or even sub-second speeds.

Numbers are similar for conversation creation: Humans averaged between 280 and 513 seconds, whereas most LLMs finished in under 30 seconds (though some larger models, such as GPT-5, occasionally took up to 100 seconds).

5.5.2 Humans vs. LLMs: created content quality

5.5.2.1 Humans are consistent

Scores measured across tasks and languages showcased an interesting phenomenon: human output quality (as measured by other humans) was remarkably consistent. In most metrics, humans achieved ratings very close to 4.5 with little deviation.

Figure 5-31. Scores by metric and language: humans

One exception was normalization quality, where human creators achieved a lower score (around 4.0) in Simplified Chinese, Kinyarwanda, and Tamil. Our analysis indicates that 1–2 human creators in the pool for those languages misunderstood certain aspects of the task, leading to lower-than-expected scores (specifically, a higher incidence of ratings 1–3). This illustrates a well-documented challenge in large-scale data annotation: even with training protocols in place, some proportion of annotators may develop incomplete or inconsistent interpretations of complex task requirements. While mitigation strategies exist, achieving uniform annotator alignment remains difficult and, in many cases, infeasible.

Other outliers were the metrics that were easier to score well: coherence and specified domain fit. Both LLMs and humans consistently scored highly on these metrics; therefore, we placed less weight on these metrics as differentiators.

When considering more complex metrics – such as grammar, naturalness, or translation accuracy – humans achieved unexpectedly consistent scores. This is especially notable given that the pools of both creators and annotators consisted of only 5–8 people each, which usually exposes scores to drift and skew based on the performance of a single individual. The consistency of human scores suggests that our grading rubric was well understood across languages, and that the human scores represent a reliable baseline for single-pass human quality at scale.

5.5.2.2 Humans produce varied content

Variability is a cornerstone of effective large-scale data generation. Merely producing slight structural variations with low lexical richness yields data of limited utility for AI training. Our analysis confirms that human creators produced highly varied content in both text generation tasks, as measured by cosine similarity and type-token ratio metrics.

Furthermore, should there be any issues with low variability of human generated data, the issue can be effectively mitigated by scaling the workforce to include a larger and more diverse pool of creators. It is

noteworthy that, while LLMs theoretically possess a vocabulary vastly superior to that of any human, the active subset of words they prioritize during generation is, in practice, significantly more constrained.

5.5.2.3 Humans are competitive – in the right places

Even though human creator scores were consistent and relatively high, they lagged the best-performing models even in long-tail languages. Humans did not rank 1st overall in any language; they were closest in Kinyarwanda at 4.50 points, trailing Gemini 2.5 Pro by 0.06 points. In many well-supported languages, such as English, French, or Polish, humans ranked last.

While these results may appear to reflect unfavorably on human content creation, the differences are negligible in many languages. Our findings demonstrate the rapid increase in the multilingual capabilities of LLMs. These results suggest a need to re-evaluate where to most effectively integrate humans in the loop.

The localization industry has long utilized post-editing workflows, where translators review and edit machine-translated segments. Results from this study suggest that LLMs may redefine workflows in broader forms of content creation, whether for marketing, legal, education, or technical writing. In terms of pure linguistic proficiency and content manipulation, SOTA models can now provide effective first drafts for humans to review, edit, and adapt – even in long-tail languages.

5.6 Variability assessment

To further evaluate the quality of outputs from both LLMs and humans, we implemented a statistical variability assessment. The objective was to determine whether any models produced content with highly repetitive vocabulary, which would render them unsuitable for text generation at scale.

5.6.1 Variability metrics used

Our study covers languages with vastly different levels of available linguistic resources (e.g., quality of taggers or semantic embedding models). To ensure fair comparison, we selected statistical measures that are independent of external tool quality.

Note, these metrics reflect performance within a specific language. Due to fundamental structural differences between languages, we recommend interpreting results separately for each language rather than making cross-language comparisons.

- **Cosine similarity (TF-IDF):** This metric measures the cosine of the angle between two text vectors. A value closer to 1 indicates high similarity (repetitiveness), while a lower value indicates diversity. We used a TfidfVectorizer to build a vocabulary from all unique words in the generated samples. Each sample was represented as a vector, and we calculated the average pairwise cosine similarity for each model-language group.
- **Type-token ratio (TTR):** TTR calculates the ratio of unique words (types) to the total number of words (tokens) in the dataset. A higher TTR indicates a richer, more diverse vocabulary. Note, in this context, 'tokens' refers to linguistic words, not the sub-word tokens used by LLM tokenizers.

5.6.2 Variability results in domain-specific paragraph generation

For the domain-specific paragraph generation task, cosine similarity results were consistently well below 0.2 across the board, indicating a generally low level of repetition.

However, 2 models distinguished themselves. GPT-5 emerged as the clear leader, ranking 1st in 5 out of 8 languages and 3rd in French. Claude Sonnet 4.5 also performed strongly, securing a top 3 position in 4 out of 8 languages. These models appear to provide the most lexically varied outputs for this task.

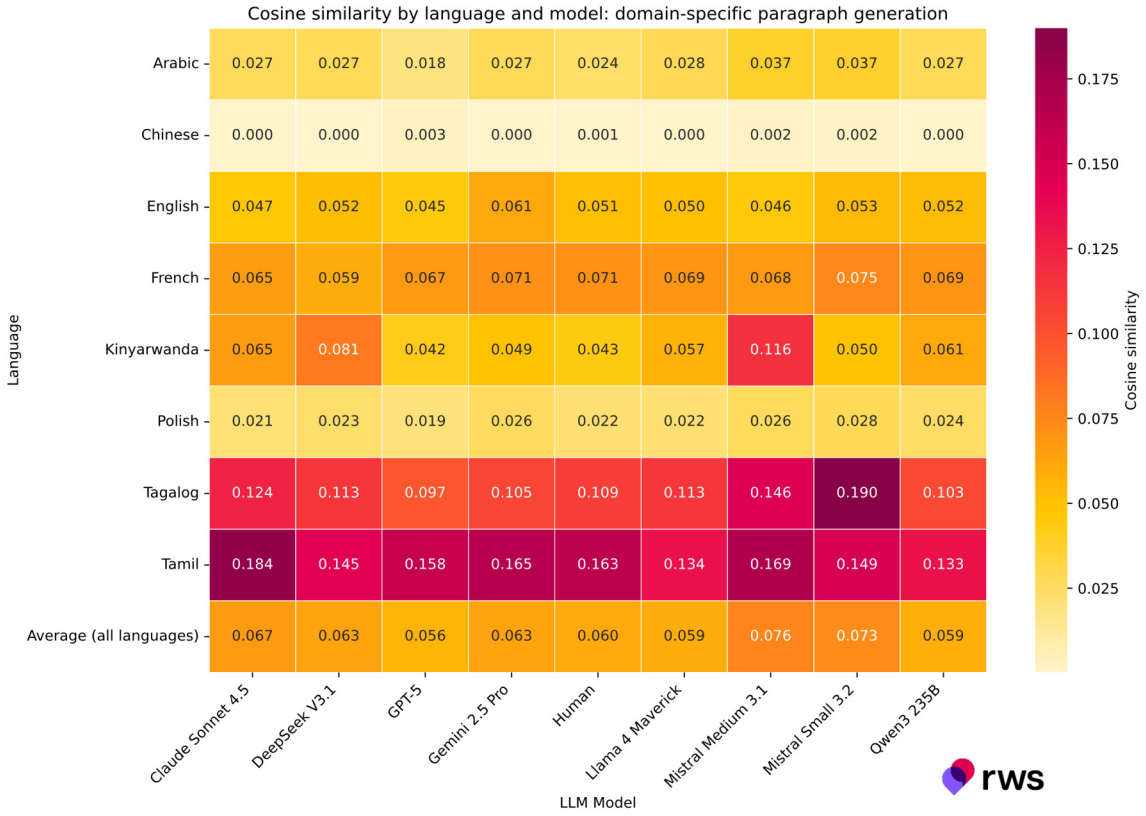

Figure 5-32. Cosine similarity by language and model: domain-specific paragraph generation

GPT-5's dominance is even more pronounced in the TTR analysis. It achieved the best result in 7 out of 8 languages and ranked 3rd in Polish. Interestingly, the human baseline was the 2nd best performer, scoring highest in Polish and placing in the top 3 in 4 other languages.

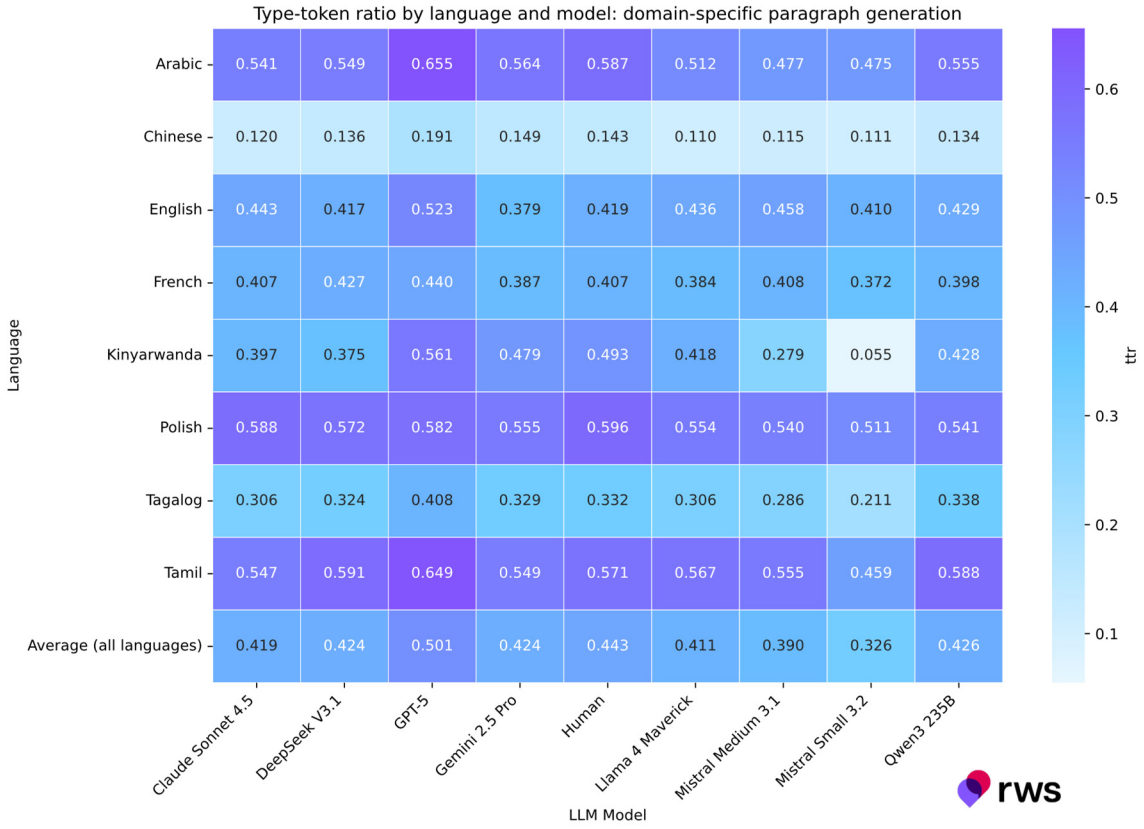

Figure 5-33. Type-token ratio by language and model: domain-specific paragraph generation

5.6.3 Variability results in conversation generation

The dominance of human-created data and GPT-5 output is equally evident in the conversation generation task. When measured by cosine similarity, human-generated content ranked highest in 6 out of 8 languages. GPT-5 followed closely, securing 2nd or 3rd place in 6 languages, reinforcing the strong performance of these 2 sources in producing naturalistic dialogue.

Figure 5-34. Cosine similarity by language and model: conversation generation

When measured by TTR variability, GPT-5 consistently produced the most diverse data across all languages. Surprisingly, the 2nd best performer was Qwen3 235B, which secured 2nd place in 6 languages and 3rd place in the remaining 2 languages.

Overall, diversity scores for conversation generation were lower than those for domain-specific paragraph generation, despite both tasks utilizing varied topic and subtopic sets. A closer examination of the data revealed that many conversations adhered to repetitive structural patterns and relied heavily on formulaic language, accounting for the disparity in lexical richness between the two tasks.

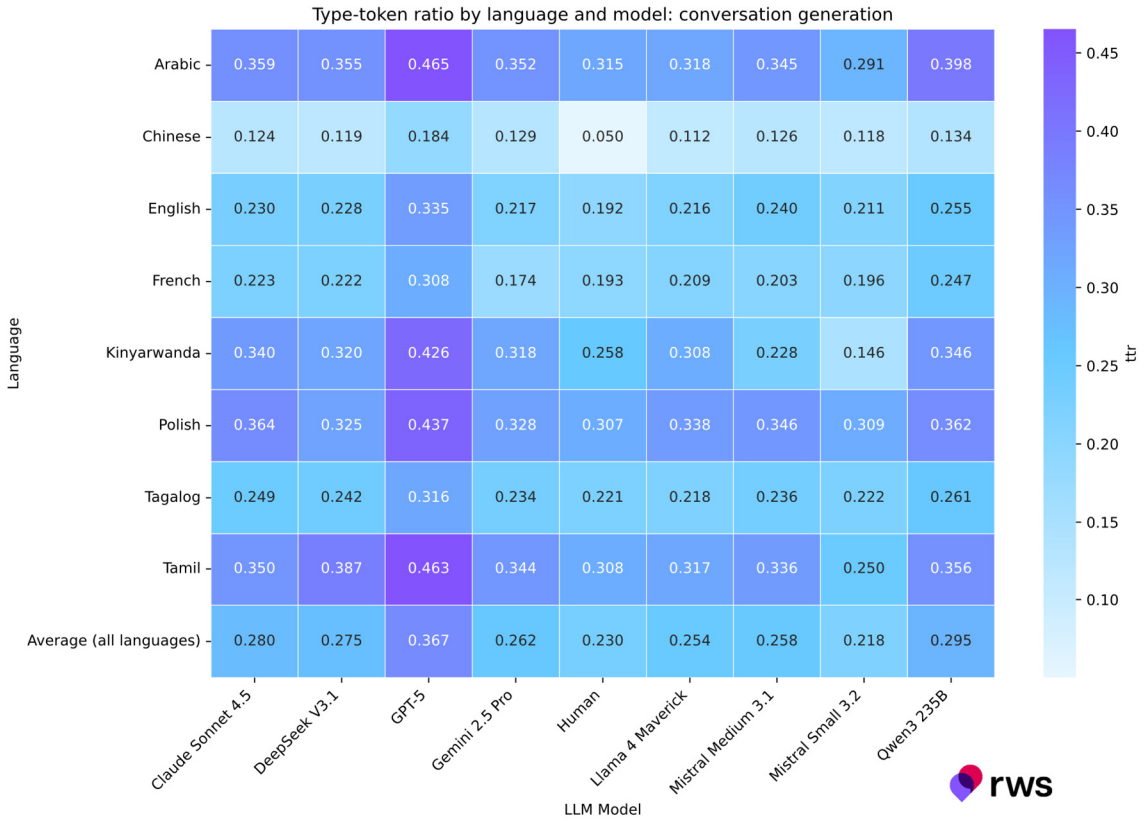

Figure 5-35. Type-token ratio by language and model: conversation generation

5.7 Data generation speed

A key finding from our previous study was the significant disparity in generation speed; for instance, we observed Llama 3.1 (405B) generating content at a pace 10x slower than GPT-4o. Such differences are critical when generating data at scale or for latency-sensitive applications (e.g., user experience, robotics).

The introduction of new architectures – particularly reasoning models – dramatically impacts "time-to-first token" and overall generation time. These models must generate a chain-of-thought (CoT) before producing the final output tokens. (See Section 5.9: Reasoning vs. non-reasoning models for a detailed discussion).

5.7.1 User throughput (characters per second)

To measure user throughput, we first measured the number of characters generated per second. This metric is valuable because it normalizes for tokenizer differences and focuses on the end-user outcome. As noted in Section 5.2.2.2: Adherence to task instructions, conversation generation, content length varies significantly across models. Unlike 'tokens per second' (a standard engineering metric), 'characters per second' reflects the practical reality for AI practitioners: it answers the question 'how fast can I get the expected volume of usable text?'

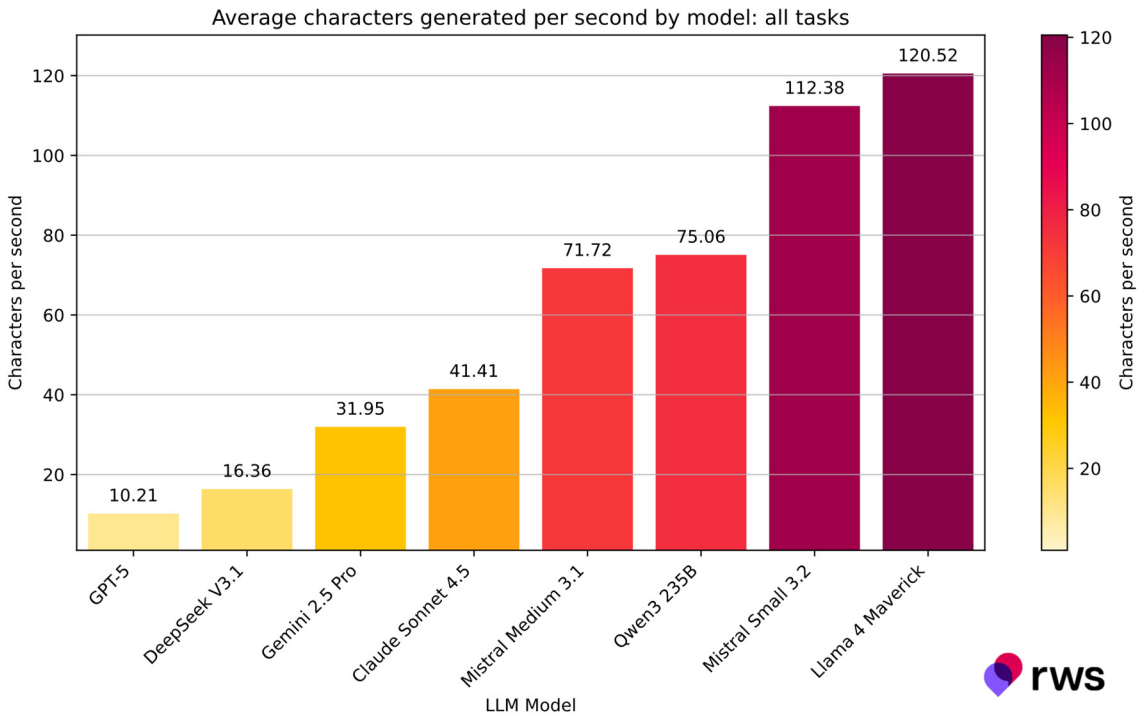

Figure 5-36. Average characters generated per second by model: all tasks and languages

When measuring user throughput using characters per second, Llama 4 Maverick emerged as the fastest model. This is surprising for two reasons: first, it outperforms the significantly smaller Mistral Small 3.2 (400B vs. 24B); second, it exceeds GPT-5's speed by a factor of 10. Interestingly, this represents a complete reversal of the findings from our previous study.

5.7.2 Raw model speed (tokens per second)

Because Llama 4 Maverick was competing against reasoning models (GPT-5 and Gemini 2.5 Pro), a direct character-speed comparison tells only half the story. We must account for the time these models spent generating reasoning tokens (which are not part of the final character output).

To assess the raw throughput capability, we measured generation speed in tokens per second, including the reasoning tokens in the total count. This demonstrates the potential speed achievable if reasoning capabilities were disabled or token budget reduced.

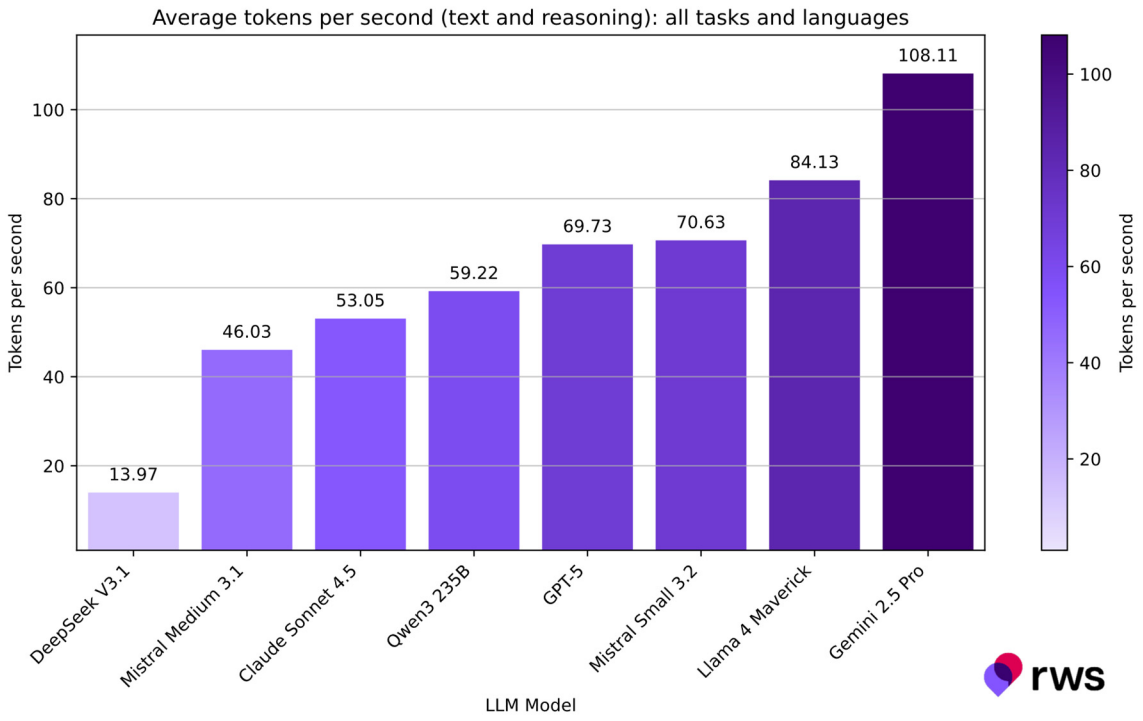

Figure 5-37. Average tokens per second (text and reasoning): all tasks and languages

When including reasoning tokens, GPT-5 and Gemini 2.5 Pro prove to be highly efficient models. In this setting, GPT-5 was only 15–20% slower than Llama 4 Maverick, and Gemini 2.5 Pro was 20% faster than the Llama model. It is safe to assume that without the reasoning overhead, these 3 models would achieve comparable generation speeds.

5.7.3 Performance outliers

Conversely, DeepSeek V3.1 showed underwhelming generation speed, processing tokens 3x slower than the next slowest model (Mistral Medium 3.1) and nearly 8x slower than the leader (Gemini 2.5 Pro). While we cannot rule out deployment-specific issues at the inference provider (Azure AI Foundry), these findings are consistent with other independent benchmarks.

While measurable differences exist among the other models, we do not consider them significant enough to be the primary factor in model selection, even for large-scale data generation.

Note, we utilized different inference providers for individual models, which can influence latency. Specific providers are documented in section 7 Model overviews.

5.8 Tokenizer efficiency

Our main metric to measure tokenizer efficiency is characters per token, which controls for the length of the generated content since it can vary significantly between models. The results of our testing are visualized in the following table (higher is better).

Figure 5-38. Characters per token by language and model: all tasks and languages

Based on our testing, OpenAI no longer has the most efficient tokenizer for the languages we tested. In our previous study, GPT 4o dominated, but this time Gemini 2.5 Pro had a significant lead with an average of 3.67 characters per token. GPT-5 finished 2nd and led in 2 languages but was about 10% less efficient than Gemini. Gemini 2.5 Pro led in most languages, and in some cases by a very wide margin: in Tamil, it outperformed the runner-up (GPT-5) by 1.34 characters per token. In other languages, the difference between the leader and the runner-up was at most 0.34 characters per token, while for Chinese, the 2 best models were tied. Gemini's performance in English also far exceeds Google's stated estimate of 4 characters per token: we measured its performance at 5.7 characters per token.

The performance of GPT-5's tokenizer was in line with our prior tests: it showed a variance of less than 0.2 characters per token compared to GPT 4o, which uses an identical tokenizer (o200k_base). The small variation is likely caused by different samples of text.

We saw a significant generational improvement in tokenizer efficiency in Mistral and Llama models. In our previous study, these models had tokenizers that were 3-4x less efficient in languages like Tamil and Arabic compared to GPT 4o. In most languages, the Mistral and Llama tokenizers are now highly competitive, with Mistral leading in Arabic. However, we continue to see poor tokenizer performance in Tamil for Llama 4 Maverick, and in Chinese for Mistral models.

Anthropic does not disclose information about its tokenizers, but Claude Sonnet 4.5 appears to be using an upgraded tokenizer compared to Claude 3.5 Sonnet, which we tested previously. We saw improvements in tokenizer efficiency particularly in English and French; tokenizer efficiency in other languages showed little to no improvement. Despite these advances, Claude's tokenizer is no match for the updated tokenizers of other models, which likely utilize larger dictionaries – it ended up ranking in last place. If token consumption or cost is a factor, we do not recommend Claude Sonnet 4.5, especially for languages that use non-Latin scripts (such as Arabic, Tamil or Chinese), where it can end up consuming up to 3.5x more tokens than other models.

The Chinese models (DeepSeek V3.1 and Qwen 3 235B) performed very similarly across all languages, with the only exception being Qwen's poor performance in Tamil. For Simplified Chinese tokenization, they ranked 1st and 3rd, respectively, although DeepSeek V3.1 tied for the win with Gemini 2.5 Pro.

5.9 Reasoning vs. non-reasoning models

In late 2024, AI labs began leveraging test-time compute, enabling models to generate an internal CoT during inference (often entirely hidden from the end user) before producing final output tokens. These systems are typically referred to as "reasoning models," though the more anthropomorphic term "thinking" is occasionally used. When combined with tool use, this reasoning approach has proven exceptionally effective at solving complex tasks, a capability reflected in their dominant performance across intelligence and problem-solving benchmarks.

This paradigm shift has significant implications for multilingual capabilities, ranging from increased token consumption and the need for tokenizer efficiency, to questions regarding the specific language used for internal reasoning and the impact of cross-lingual transfer.

5.9.1 Reasoning model selection

We adhered to default API inference settings as much as possible, intentionally leaving parameters related to "thinking" or reasoning untouched. For example, Claude Sonnet 4.5 requires an explicit parameter to activate its extended thinking capabilities that we did not utilize. Similarly, we opted against using the specialized "thinking" variant of Qwen3 235B, as there is currently no clear evidence that reasoning processes significantly enhance core linguistic proficiency.

We acknowledge that these configuration choices may have influenced scores. One might hypothesize that GPT-5's performance on text manipulation tasks was bolstered by its extensive internal CoT, particularly given that it consumed 2–4 times more reasoning tokens in text normalization than in other tasks. However, as the case of GPT-5 demonstrates, this was not a deciding factor: even with this potential advantage, it failed to secure a top 3 ranking, ultimately being surpassed by 2 non-reasoning models.

The specific performance delta between a model's standard and reasoning-enabled modes remains relatively understudied and represents a compelling avenue for future research.

5.9.2 Reasoning can be expensive

Two of the evaluated models functioned as "reasoners" by default, offering a clear window into the cost implications of this approach. GPT-5 produced particularly extensive chains-of-thought, with reasoning tokens accounting for nearly 90% of its total output volume. In practical terms, this means GPT-5 consumes roughly ten times more tokens than a non-reasoning model to produce the identical sentence.

Gemini 2.5 Pro exhibited more restraint, consuming approximately half the reasoning tokens of GPT-5. However, this still resulted in a substantial overhead – roughly 4.5 times the token consumption compared to the text token count without reasoning. It is worth noting that, for other workflows, the length of these reasoning chains and the resulting cost will likely vary based on task complexity.

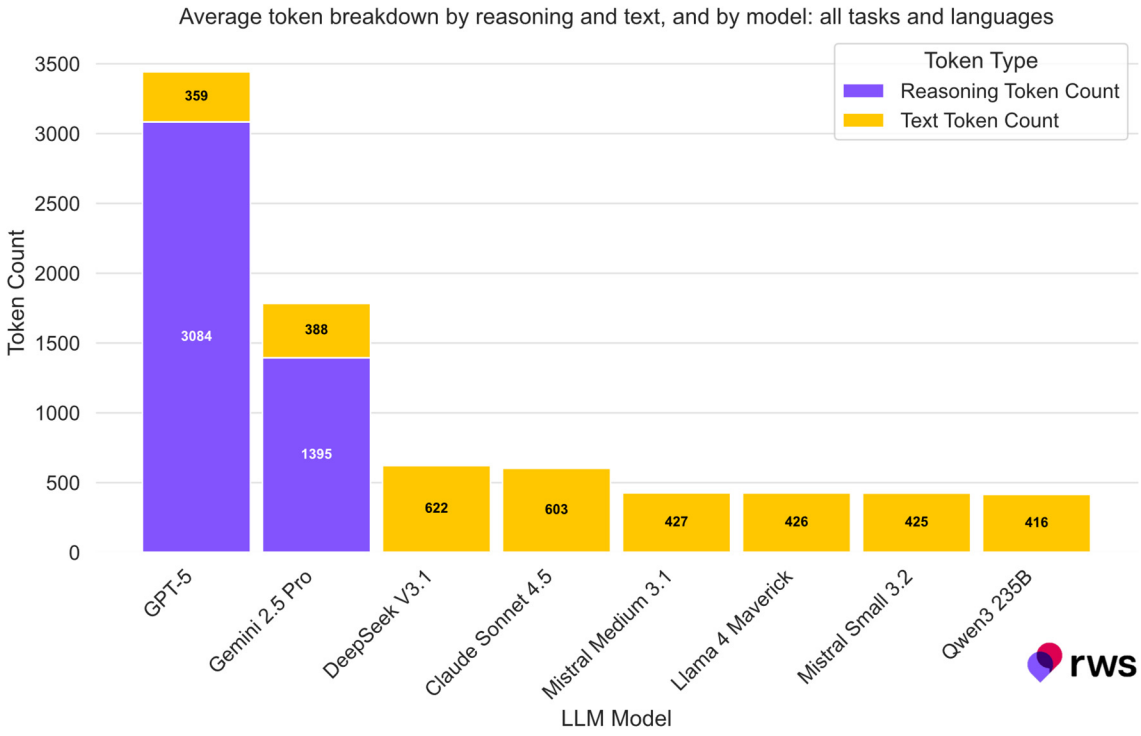

Figure 5-39. Average token breakdown by reasoning and text, and by model: all tasks and languages

Naturally, even with reasoning models, "thinking" mode can typically be disabled or constrained by setting a reasoning budget. More critically, however, it appears that these models necessitate distinct strategies for efficient large-scale data generation and manipulation. For instance, aggressive content batching within a single request emerges as a valid strategy to avoid generating expensive reasoning chains for each individual item.

Additionally, the reasoning budget represents another critical "dial" to calibrate during benchmarking or production deployment. It is essential to recognize that disabling the thinking process or severely restricting this budget may have significant negative impact on output quality, adding an additional tradeoff layer between cost and quality.

5.9.3 Tokenizer efficiency has real impact on reasoning models

We previously observed that differences in tokenizers, specifically how efficiently they encode words across various languages, have a significant impact on cost. One might argue that since 75–90% of a reasoning model's output is consumed by internal "thought," the relative importance of tokenizer efficiency is diminished. However, these factors do not cancel each other out; rather, they stack and multiply. An inefficient tokenizer will likely inflate both the final text output and the volume of reasoning tokens required to generate it.

Consider the compounding effect: the difference between a standard model with an efficient tokenizer, and a reasoning model that allocates 90% of its budget to thinking, while encoding text at only 50% efficiency, resulting in an effective 20-fold increase in token consumption. While this specific scenario is hypothetical, the risk is real, given that we observed tokenizer efficiency differences of 2x or greater in 3 of the tested languages.

5.9.4 In what languages do LLMs think?

Recent studies indicate that "many LLMs predominantly reason in English or Chinese, even when prompted in other languages" (Hwang et al., 2025). This raises significant questions regarding the cultural relevance of generated outputs, particularly when the underlying context remains anchored in English or Chinese. It is well-established that the prompting language significantly influences the cultural context of the response (Lu et al., 2025); we hypothesize that a similar effect could be at play here, where the target language content is effectively "polluted" by an English-based reasoning chain.

This phenomenon has 2 distinct implications. First, the hypothetical cost scenario discussed previously becomes more complex: when prompted in Tamil, a model might generate the bulk of its reasoning tokens in English, effectively leveraging higher tokenizer efficiency for that portion of the process. Conversely, this English-mediated reasoning introduces a theoretical risk of "cultural contamination," where the final output becomes less culturally authentic due to the misalignment between the reasoning language and the target locale. This warrants further investigation, as we have observed considerable variation across models in their selection of internal reasoning language - variation that may have measurable effects on both cost efficiency and output quality.

5.10 Performance preview: Gemini 3 Pro

During the data generation and annotation phases of this study, the Gemini 3 Pro preview was released on November 18, 2025. Given that our preliminary results indicated that Gemini 2.5 Pro was poised to be a top contender (if not the overall leader), we were eager to evaluate its successor. However, retroactively generating and annotating Gemini 3 Pro's outputs would have violated our study standards, specifically regarding full randomization, and introduced the risk of rating and calibration drift.

As a partial solution, we integrated Gemini 3 Pro Preview's outputs into the final 2 tasks where annotation had not yet commenced. We excluded these results from the main analysis to prevent missing data from skewing cross-task aggregations. Instead, we dedicate this separate section to observations related to Gemini 3 Pro Preview, focusing specifically on how it compares to its predecessor, Gemini 2.5 Pro. Please note that the final (non-preview) version of the model may exhibit slightly different behaviors.

5.10.1 Issues generating translations

Text translation exposed significant stability issues in Gemini 3 Pro Preview in relation to structured outputs, as the model consistently generated unstructured plaintext responses despite strict schema requirements. This behavior proved particularly resistant to correction in French and Tamil, where valid structured outputs remained elusive despite more than 100 retry attempts per language. Even explicit prompt reinforcement failed to rectify the issue.

Consequently, we were forced to implement a custom parsing solution to extract translation pairs directly from the raw text, effectively bypassing the JSON schema requirement entirely. This represents a notable limitation, demonstrating that certain models may be fundamentally resistant to structured output constraints in specific contexts, regardless of prompt engineering. Such cases necessitate architectural workarounds that introduce additional processing complexity and increase the risk of downstream errors in the data generation pipeline.

Considering that Gemini 3 Pro was evaluated in preview mode shortly after its release, we predict that these structural issues will be resolved in future iterations, especially given that Gemini 2.5 Pro was not affected. Nevertheless, we document this behavior for the sake of completeness and to serve as a reference should these limitations persist in the general availability release.

5.10.2 Text normalization performance

Gemini 2.5 Pro delivered strong performance on the text normalization task, achieving an average rating of 4.61 across languages, though it narrowly conceded the top rank to GPT-5 (4.62). The transition to Gemini

3.0 Pro Preview demonstrates a significant generational uplift: the model achieved an overall score of 4.75, securing the highest ranking on text normalization in every language tested.

Notably, except for Kinyarwanda, Gemini 3 Pro Preview's scores on text normalization across all languages exceeded 4.74. This near-perfect performance suggests that text normalization is approaching saturation as a benchmark for frontier models, posing little challenge to systems of Gemini 3 Pro Preview's caliber. However, such tasks may retain value as a diagnostic tool for gauging processing proficiency in low-resource or rarer languages.

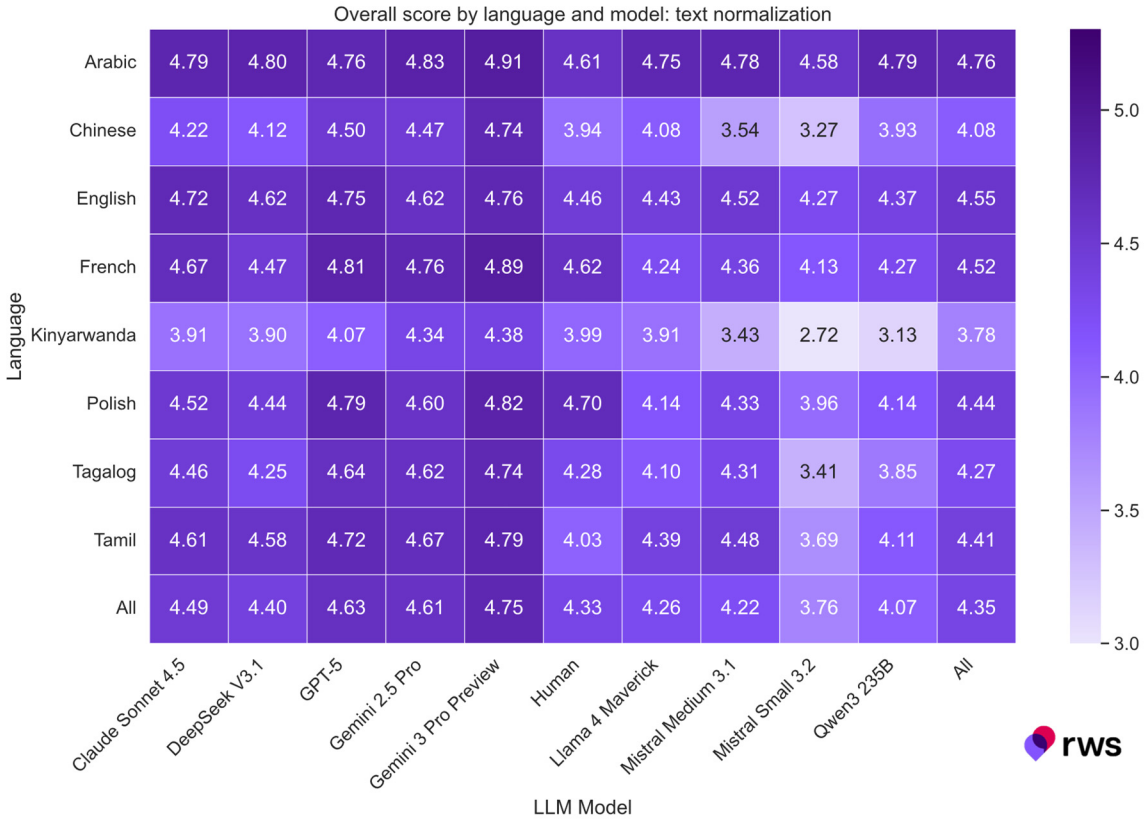

Figure 5-40. Overall score by language and model: text normalization

5.10.3 Translation performance

On the translation task, Gemini 2.5 Pro had already established dominance with a top score of 4.73, surpassing GPT-5 (4.59). Consequently, the generational improvement seen in Gemini 3 Pro Preview was more incremental, resulting in an overall score of 4.75.

This trend was mirrored in the translation accuracy metric, where Gemini 2.5 Pro scored 4.71 compared to Gemini 3 Pro Preview's 4.74. Remarkably, Gemini 3 Pro Preview did not receive any low ratings (1 or 2) for accuracy and is the sole model to achieve a score above 4.5 for translation across all languages tested.

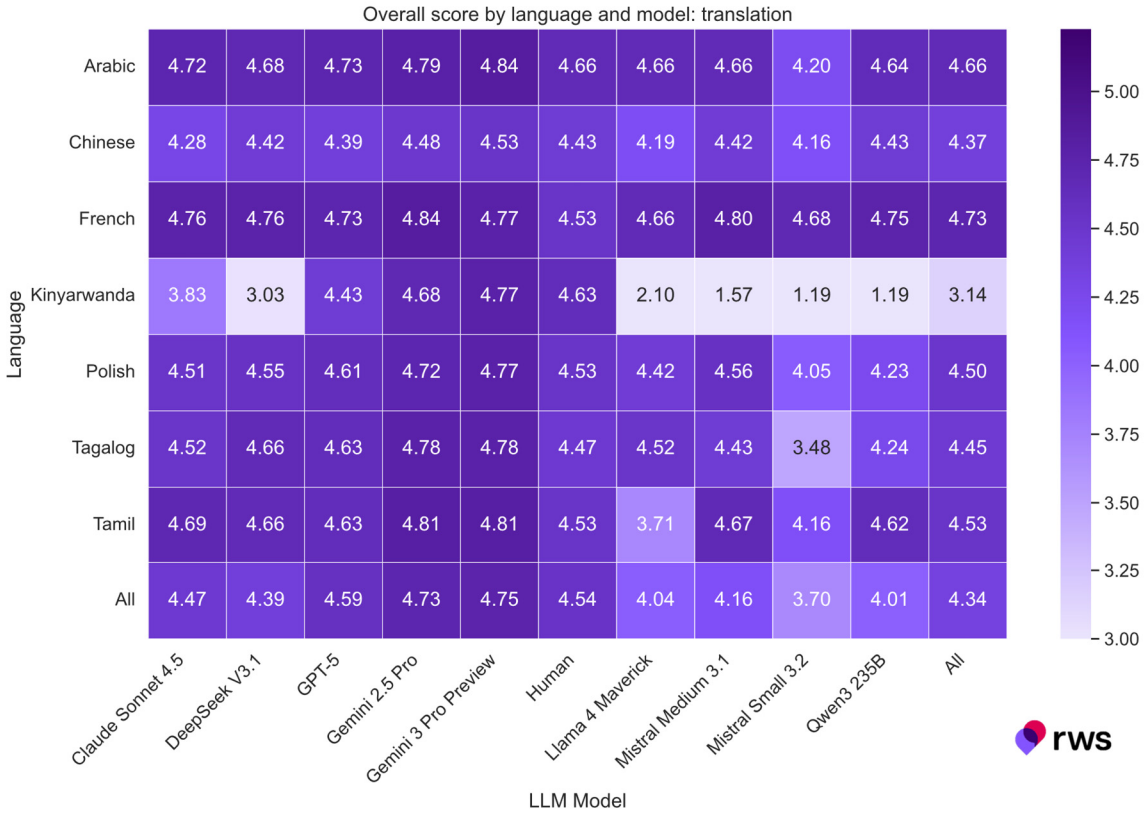

Figure 5-41. Overall score by language and model: translation

5.10.4 Tokenizer efficiency

Gemini 3 Pro Preview appears to use the same tokenizer as Gemini 2.5 Pro, as it scored identically in our tests. See Gemini's performance in Section 6.8: Tokenizer efficiency.

5.10.5 Increased reasoning token count from Gemini 2.5 Pro

One negative trend observed when comparing Gemini 2.5 Pro and Gemini 3 Pro Preview is the increase in reasoning token consumption. While Gemini 3 Pro Preview still utilizes significantly fewer reasoning tokens than GPT-5, it registered an increase of nearly 50% compared to its predecessor. We hypothesize that, at least for text manipulation tasks, the performance gains observed in Gemini 3 Pro Preview relative to its predecessor may be partially attributable to this increased test-time compute allocation.

This shift has clear implications for both cost and speed. In our benchmarks, Gemini 3 Pro Preview was the slowest model as measured by output characters per second. However, this sluggishness – driven largely by unusually high time-to-first-token (TTFT) latency – may be attributable to technical inefficiencies inherent in the preview environment, which are likely to be resolved in the general availability release.

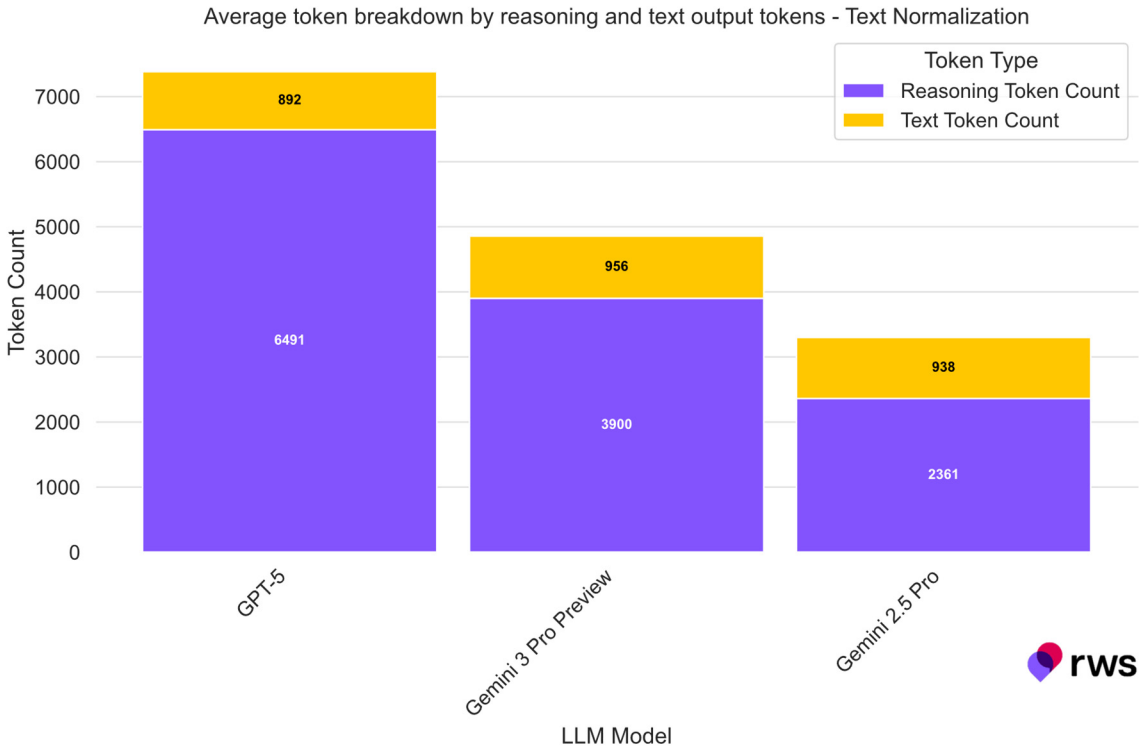

Figure 5-42. Average token breakdown by reasoning and text, and by model: text normalization

6 Model performance summaries

6.1 Claude Sonnet 4.5

Model provider	Anthropic
Model name	Claude Sonnet 4.5
Model version	eu.anthropic.claude-sonnet-4-5-20250929-v1:0
Release date	September 29, 2025
Parameters	N/A
Reasoning	Yes (Hybrid)
Distribution	Proprietary
Supported languages	200+ (no specific list provided)
Inference provider	AWS Bedrock
Overall score rank	2

ANTHROP\C

Claude Sonnet 4.5 entered this iteration of the study defending the title secured by its predecessor (Claude 3.5 Sonnet), which previously topped the rankings by a 0.12-point margin over GPT-4o. The results confirm Anthropic's strong position in the foundational AI landscape: the model took first place in English and secured scores within the margin of error of top models in French, Polish, and Tamil.

That said, Claude Sonnet 4.5 trails the overall leader, Gemini 2.5 Pro, by 0.12 points. Its aggregate score was notably impacted by weaker performance in Kinyarwanda, suggesting that Anthropic's training regime prioritizes high-resource languages over the extreme long tail (despite claims of supporting 200+ languages). While there are encouraging signs, such as a respectable 4.37 in Kinyarwanda domain-specific paragraph generation, it fell below the usability threshold of 4.0 in other tasks for that language.

Regarding speed, Claude Sonnet 4.5 (inferred via AWS Bedrock) was measurably slower than most competitors. While it appears faster than GPT-5 and Gemini 2.5 Pro in raw numbers, this result is deceptive: those models include time-intensive chain-of-thought (CoT) reasoning. If reasoning were disabled on the competitor models, Claude would be slower than both, surpassing only DeepSeek V3.1. While its speeds are not critically low, Claude is far from the optimal choice if latency is a primary constraint.

A significant weakness of Claude Sonnet 4.5 lies in its tokenizer. Although Anthropic appears to have improved upon the 3.5 version, substantial gains in the characters per token metric were observed only in English (3.47 → 4.80) and French (2.80 → 3.33). In other languages, improvements were negligible (< 0.5 chars/token).

Compared to the field, Claude Sonnet 4.5 utilizes the least efficient tokenizer, ranking last in 6 out of 8 languages. The disparity is starkest in Arabic (less than 50% of Gemini 2.5 Pro's efficiency) and Tamil (less than 30%). We strongly advise practitioners deploying Claude Sonnet 4.5 to global audiences to weigh this operational cost heavily.

We consider Claude Sonnet 4.5 a highly competitive model that is beginning to show proficiency in long-tail languages. However, it is hampered by subpar inference speed and significant tokenizer inefficiencies in multilingual settings.

6.2 DeepSeek V3.1

Model provider	DeepSeek
Model name	DeepSeek V3.1
Model version	DeepSeek-V3.1
Release date	August 28, 2025
Parameters	671B
Reasoning	Yes
Distribution	Open-weights
Supported languages	100+ (no specific list provided)
Inference provider	Azure AI Foundry
Overall score rank	3

DeepSeek V3.1 presents a compelling case from 2 distinct perspectives: it is a relatively new entrant to the foundational AI race, and it stands as the largest open-weights model we tested. This characteristic makes it uniquely suitable for organizations prioritizing customization or requiring deployment within their own infrastructure.

From an overall quality perspective, the model delivered robust performance, securing 3rd place overall behind Gemini 2.5 Pro and Claude Sonnet 4.5. It achieved scores exceeding 4.5 in 7 out of 8 languages, demonstrating remarkable consistency across tasks and metrics (with Kinyarwanda being the only notable exception). Consequently, for many standard applications, it serves as our recommended choice for budget-conscious projects, offering an excellent trade-off between cost and quality.

However, operational reliability presents a concern. During output generation, we encountered recurring difficulties with adherence to the requested JSON schema. Specifically, for the text normalization task, we were forced to abandon our standard bulk-processing strategy in favor of individual requests to ensure successful execution. This phenomenon suggests that while the model can generate high-quality text outputs, it may not yet be sufficiently reliable for complex workflows that depend heavily on strict formatting and structured data integration.

Inference speed is another significant limitation. DeepSeek V3.1 was by far the slowest model in our roster, generating outputs roughly 8x slower than the fastest competitors. This significant latency renders the model largely unsuitable for applications requiring real-time responsiveness or high-volume data generation. Furthermore, its tokenizer was measurably less efficient, particularly in Tamil, which introduces additional latency and cost considerations for global deployments.

DeepSeek V3.1 remains a highly capable model, signaling that the lab is on a clear trajectory towards achieving SOTA status. It currently stands as one of the most competitive budget options available.

However, its viability for scaled enterprise settings is tempered by specific limitations. The combination of slow generation speeds and occasional instability with structured outputs poses significant challenges for production workflows, though we expect these issues to be addressed in future iterations.

6.3 Gemini 2.5 Pro

Model provider	Google
Model name	Gemini 2.5 Pro
Model version	gemini-2.5-pro
Release date	June 17, 2025
Parameters	N/A
Reasoning	Yes
Distribution	Proprietary
Supported languages	116 (specific list)
Inference provider	Gemini Developer API
Overall score rank	1

Gemini had already demonstrated significant potential with its 1.5 Pro release; in our previous study, we observed performance on par with GPT-4o and noted its superiority in translation tasks. With the release of Gemini 2.5 Pro, Google has achieved a decisive leap forward, securing a commanding victory in the current iteration of our study. The model achieved an aggregate score of 4.73, distinguishing itself as the only model to score above 4.5 in every language tested, including the challenging, long-tail language of Kinyarwanda. In fact, the model's performance was so robust that it effectively saturated our current benchmarks, necessitating an increase in task complexity for future iterations of this study to effectively differentiate top-tier performance.

The consistency of Gemini's scores across all languages, tasks, and metrics positions it as the premier choice for broad multilingual support. Crucially, this marks a paradigm shift in LLM capabilities. Until recently, LLMs were considered largely unusable for long-tail languages, but Gemini 2.5 Pro fundamentally challenges this assumption by not only scoring well in Kinyarwanda and Tamil but also surpassing our single-pass human baseline in both.

Our evaluation revealed no significant operational downsides. In terms of raw throughput (tokens per second), Gemini leads the field. While its characters-per-second metric appeared lower, this was solely due to the generation of extensive CoT reasoning tokens, a feature that can be disabled to further accelerate output. Furthermore, Gemini's tokenizer efficiency is currently industry-leading, surpassing even that of GPT-5.

A minor limitation, however, was the measurably lower lexical variability of Gemini 2.5 Pro's outputs, particularly in the conversation generation task. This stands in stark contrast to our previous study, where Gemini 1.5 Pro excelled in variability; however, given the differences in methodology, these results are not directly comparable.

Gemini 2.5 Pro is our top recommendation for any multilingual use case, ranging from simple text generation to complex manipulation tasks. Additionally, our testing of Gemini 3 Pro Preview on text manipulation tasks indicates measurable improvements even over the 2.5 baseline. Together, these models establish a new standard for fast, efficient, and truly multilingual foundational AI.

6.4 GPT-5

Model provider	OpenAI
Model name	GPT-5
Model version	gpt-5-2025-08-07
Release date	August 7, 2025
Parameters	N/A
Reasoning	Yes
Distribution	Proprietary
Supported languages	N/A
Inference provider	OpenAI Platform
Overall score rank	4

In our previous study, OpenAI's GPT-4o and its Mini variant performed exceptionally well, with the full model securing 2nd place in the overall ranking. It consistently achieved high scores across most languages, with the notable exceptions of Tamil and Kinyarwanda, regardless of task type. Consequently, expectations for OpenAI's newest generation of models were substantial.

In our current tests, GPT-5 delivered decidedly mixed results. It demonstrated significant strength in text manipulation, ranking 1st in text normalization and 2nd in translation. However, its performance on text generation tasks was less impressive, ranking 6th in conversation generation and tying for 6th in domain-specific paragraph generation with Qwen3 235B. Moreover, performance was highly inconsistent across languages: while it scored reasonably well in English, French, Arabic, and Tagalog for text generation, its outputs in Chinese and Polish were underwhelming, frequently dipping below the 4.0 usability threshold. Given its strong results on manipulation tasks, GPT-5 appears proficient in understanding numerous languages but struggles with original content creation, often producing unnatural or repetitive text. This disparity may result from optimization strategies targeting benchmarks heavily skewed toward coding, logic puzzles, and analytical reasoning. Regardless of the root cause, we caution against relying on GPT-5 for synthetic multilingual text generation due to this instability.

Regarding operational metrics, data generation was slower overall, similar to Gemini 2.5 Pro, due to the latency introduced by long CoT reasoning. However, the raw token-per-second throughput was perfectly acceptable, suggesting that if reasoning were disabled, generation speeds would be competitive.

GPT-5's tokenizer ranked 2nd in efficiency, trailing only Gemini 2.5 Pro, with negligible differences observed in most languages. A key exception, however, was Tamil, where Gemini 2.5 Pro's tokenizer remained roughly 30% more efficient – a significant factor for highly multilingual applications where cost and latency are critical.

GPT-5 demonstrated superior output variability, outperforming all other models in both TTR and cosine similarity metrics. In this regard, its only true competition came from human creators, who remained the sole challengers to its lexical diversity.

GPT-5 is a capable model that is beginning to demonstrate proficiency even in long-tail languages. However, due to the high instability of its results depending on the specific task and language, we strongly recommend comprehensive testing across the entire target language scope before committing to this model for production environments.

6.5 Llama 4 Maverick

Model provider	Meta
Model name	Llama 4 Maverick
Model version	Llama-4-Maverick-17B-128E-Instruct-FP8
Release date	April 5, 2025
Parameters	400B
Reasoning	No
Distribution	Open-weights
Supported languages	12 (specific list)
Inference provider	Azure AI Foundry
Overall score rank	5

Llama 4 Maverick, the 2nd largest open-weights model in our roster, marks the return of the Llama family. In our previous study, Llama 3.1 405B was noted less for its top-tier scores and more for its consistency, outperforming Claude 3.5 Sonnet by maintaining scores above 4.25 in all languages except Kinyarwanda. Llama 4 Maverick retains this signature trait, repeating the achievement of scoring above 4.25 in all languages excluding Kinyarwanda.

However, the competitive landscape has shifted. While Llama 4 Maverick decisively outperformed Qwen3 235B and Mistral Small 3.2, and technically edged out GPT-5 and Mistral Medium 3.1 (within the margin of error), it now trails significantly behind the market leaders. Overall, it fell 0.28 points short of Gemini 2.5 Pro. While it demonstrated strong capability in Tamil (4.52), its performance in Kinyarwanda remains underwhelming (3.65).

That said, we observed significant improvements over the Llama 3.1 family. Llama 4's tokenizer is now relatively competitive, averaging 2.97 characters per token (compared to Gemini's 3.67), though it still lags noticeably in Tamil. Additionally, its generation speed, which was a major weakness of previous Llama models, is now excellent, ranking either 1st or 2nd depending on the specific metric applied.

Regarding reliability, we encountered some friction with Llama's ability to strictly adhere to predefined JSON schemas, though these issues were far less severe than those observed with DeepSeek V3.1.

Llama 4 Maverick serves as a viable alternative for organizations where hosting models on custom infrastructure is non-negotiable, particularly if the speed limitations and structured output instability of DeepSeek V3.1 are deal-breakers. However, for most other use cases, the current foundational AI landscape offers superior options.

6.6 Mistral Medium 3.1

Model provider	Mistral
Model name	Medium 3.1
Model version	mistral-medium-2508
Release date	August 12, 2025
Parameters	N/A
Reasoning	No
Distribution	Proprietary
Supported languages	N/A
Inference provider	Mistral Le Plateforme
Overall score rank	6

Mistral Medium 3.1 delivers solid but not state-of-the-art performance, consistent with its mid-tier positioning. The upcoming Mistral Large 3 is expected to challenge top-tier models. Mistral Medium 3.1 generally managed to score above 4.5 across tasks and languages, except for Chinese (which was one of Mistral's strong languages in our previous study), and Kinyarwanda in which it predictably failed. Its score on French (4.73) was very good, but given the strength of other current generation SOTA models, it only secured 4th place overall. We consider Mistral Medium 3.1 to be in the same model tier as DeepSeek V3.1 and Llama Maverick, although unlike the other models (and its smaller counterpart), Mistral Medium 3.1 is not open-weights.

Similarly to Llama 4 Maverick, Mistral Medium 3.1's tokenizer also saw improvements over the 2nd Mistral model generation. Its score of 3.25 characters per second is very competitive with Gemini 2.5 Pro's top score of 3.67, and Mistral achieved decent tokenizer efficiency scores even on Tamil, where most models struggled.

Regarding speed, Mistral Medium 3.1 displayed below average speeds of data generation, ranking 4th or 5th, depending on metric. This is a potential factor for use cases with low latency needs, but should not be a reason to avoid this model in most other use cases.

Mistral Medium 3.1 demonstrates some good multilingual capabilities with solid scores, but falls slightly short of SOTA model performance. It's a viable option for budget-conscious scenarios, but the budget-friendly space is already saturated, both by other models we tested (such as DeepSeek V3.1 or Llama 4 Maverick) and by several models beyond the scope of this study.

6.7 Mistral Small 3.2

Model provider	Mistral
Model name	Small 3.2
Model version	mistral-small-2506
Release date	June 20, 2025
Parameters	24B
Reasoning	No
Distribution	Open-weights
Supported languages	N/A
Inference provider	Mistral Le Plateforme
Overall score rank	8

Mistral Small 3.2 is the smallest model in our roster, requiring an adjustment of expectations regarding its multilingual capabilities. Our closest benchmarks from the previous study were Llama 3.1 70B and Llama 3.1 8B; notably, Mistral Small's performance tracked closely with Llama 3.1 70B. This is a significant achievement: despite being roughly 1/3 the size of Llama 3.1 70B, it delivers comparable multilingual utility, highlighting distinct efficiency gains in this generation of architectures.

Predictably, the model struggled with Kinyarwanda, receiving poor scores below 3.0. However, it scored 4.00 in Tamil and Tagalog, placing it just on the threshold of usability. Conversely, it excelled in high-resource languages, scoring particularly well in English (4.69) and French (4.67), with robust results in Arabic (4.52) and Chinese (4.36). In Polish, while its score was relatively modest at 4.16, it managed to outperform the significantly larger Qwen3 235B model by a small margin.

A clear divergence in performance appears between task types. Mistral Small 3.2 performed better on text generation tasks than on text manipulation tasks. This is expected, as its smaller parameter count limits the "intelligence" required for complex manipulation instructions. However, for lighter generation tasks, it remains highly capable across a wider range of languages; for instance, it scored 4.47 in Tamil and 4.82 in Chinese for domain-specific paragraph generation.

Operationally, the model uses the same competitive tokenizer as Mistral Medium 3.1, performing efficiently even on complex scripts like Tamil. Its generation speed was very good, as is expected for a model of its size, though it was surprisingly outperformed by the much larger Llama 4 Maverick. This discrepancy could be attributed to deployment differences (we utilized Mistral's La Plateforme for inference of Mistral Small 3.2) or architectural efficiencies in Llama. Regardless, both its tokenizer efficiency and generation speed remain highly competitive.

Mistral Small 3.2 exemplifies the rapid evolution of small-form-factor models, providing sufficient linguistic proficiency in a growing number of languages. While it does not rival SOTA models and still struggles with the long tail, it demonstrates a clear generational uplift.

Uniquely, it is the only model in our test group capable of running on consumer hardware, making it an excellent candidate for local deployment, experimentation, and specialized fine-tuning.

6.8 Qwen3 235B

Model provider	Alibaba
Model name	Qwen 3 235B
Model version	qwen.qwen3-235b-a22b-2507-v1:0
Release date	July 21, 2025
Parameters	235B
Reasoning	No
Distribution	Open-weights
Supported languages	119 (specific list)
Inference provider	Azure AI Foundry
Overall score rank	7

Qwen3 235B, Alibaba's open-weights model, positions itself as a robust multilingual LLM with support for 119 languages. Our testing confirmed this broad proficiency, with the model achieving scores of 4.00 or higher in 7 out of the 8 languages tested. However, in terms of ranking, Qwen3 235B typically trailed Llama 4 Maverick and Mistral Medium 3.1 by an overall margin of 0.1 points, consistently outperforming only the much smaller Mistral Small 3.2. Even in Chinese, where a local model might be expected to dominate, its respectable score of 4.47 fell short of the SOTA model benchmark set by Gemini 2.5 Pro (4.68). Notably, the model was particularly strong at domain-specific paragraph generation, suggesting it is a viable candidate for simpler synthetic data generation tasks across a wide linguistic scope.

Operationally, Qwen3's generation speed was adequate; while it did not match the high throughput of Llama 4 Maverick or Mistral Small 3.2, it performed sufficiently enough to avoid becoming a bottleneck. However, tokenizer efficiency proved to be a significant liability. The model ranked 7th on this metric, with notable degradation on complex scripts. In Tamil, for instance, it averaged only 0.89 characters per token - representing nearly a 5x efficiency deficit compared to Gemini 2.5 Pro's 4.24.

Ultimately, the performance profile of Qwen3 235B aligns with its size: it comfortably surpasses Mistral Small 3.2 but fails to bridge the gap to Llama 4 Maverick or Mistral Medium 3.1. While the model offers utility, particularly for specific scopes of well-represented languages or simpler generative tasks, the current foundational AI landscape offers several alternatives that deliver equal or superior performance at similar size and cost.

Find out more.

rws.com/trainai

About us

RWS is a global AI solutions company empowering the world's most trusted enterprise AI.

Our proprietary Cultural Intelligence Layer, powered by 250,000 data specialists, cultural and language experts and deep domain professionals, backed by 45+ patents, makes enterprise AI culturally fluent, contextually accurate and secure, ensuring every interaction reflects a brand's tone, context and customer values.

Through our Generate, Transform and Protect segments, we deliver intelligent content, enterprise knowledge, large-scale localization and IP protection for global growth. Trusted by 80+ of the world's top 100 brands, RWS provides the confidence, governance and expertise organizations need to deploy AI safely, responsibly and at scale.

Headquartered in the UK, RWS is listed on AIM (RWS.L).

More information: rws.com.

Hello, world.