

Do LLMs Cross Alphabets? A Cross-Script Audit of LLMs Across Four Kazakh Latin Proposals

Galamaddin Askar,^{1,2,3} Arailym Mukhametkali,^{1,2} Akylbek Maxutov^{1,3*}
Zhanat Makhataeva,^{1,3} Nūrali Medeu,¹ Huseyin Atakan Varol^{1,2,3}

¹Institute of Smart Systems and Artificial Intelligence, Astana, Kazakhstan

²School of Computing and Artificial Intelligence, Nazarbayev University, Astana, Kazakhstan

³Department of AI & Big Data, Al-Farabi Kazakh National University, Almaty, Kazakhstan
{akylbek.maxutov, issai}@nu.edu.kz

Abstract

Kazakhstan is transitioning the Kazakh language from Cyrillic to a Latin alphabet, with four official Latin proposals issued since 2017. How well do current large language models, trained almost entirely on Cyrillic Kazakh, handle this script change? We evaluate 13 models spanning general-purpose open-weight, Kazakh-adapted, and proprietary systems on three Kazakh benchmarks (KazMMLU, KazCulture, GSM8k-Kazakh) rewritten in all four Latin variants. The mean accuracy drop from Cyrillic to Latin is 16.8 points, ranging from 1.4 (GPT-5) to 30.4 (Sherkala-8B). Contrary to the natural tokenization-explosion hypothesis, fertility analysis shows that Latin Kazakh requires no more tokens per word than Cyrillic for most tokenizers. Subword fragmentation therefore contributes to the drop but does not primarily explain it. Kazakh-adapted models drop hardest, suggesting that Cyrillic-only adaptation entrenches script-specific routes rather than building script-agnostic competence. Prompt-level mitigations (hint, mapping) recover only 1 point on average. On the other hand, LoRA fine-tuning recovers up to +20 points, but only for sufficiently large reasoning-capable bases. Generative AI will play a key role in the script transition. Models serving Kazakh users must reliably process Latin script, which our findings show cannot be achieved at inference time. Addressing this gap requires incorporating Latin Kazakh data during pretraining or large-scale continued pretraining, rather than relying solely on prompt or adapter-level solutions.

1 Introduction

Writing systems can change over time. Throughout the twentieth century, there were cases in which a nation changed the alphabet of its national language to address phonological fit, improve literacy, pursue modernization, or align culturally and geopolitically. For example, Türkiye replaced the Ot-

toman Persian-Arabic alphabet with a Latin one in 1928 as part of reforms (Lewis, 1999). Azerbaijan voted to return from Cyrillic to Latin in 1991 and completed the transition a decade later (Hatcher, 2008). Uzbekistan adopted a Latin alphabet by law in 1993, has revised it multiple times, and remains in transition (Uzman, 2010). These reforms spanned decades and required several iterations of the new alphabet.

Kazakhstan is undergoing a similar transition. Kazakh, the state language of the Republic of Kazakhstan, is spoken by 13 to 16 million people worldwide (Koto et al., 2025; Togmanov et al., 2025). Over the past century, three alphabets have been used: Persian-Arabic until 1929, Latin-based Yañalif from 1929 to 1940, and Cyrillic from 1940 onward (Bekzhanova and Makoelle, 2022). A 2017 presidential decree began a transition back to a Latin alphabet, initially targeting 2025. The proposed alphabet has been revised three times (2018, 2019, 2021).

At the same time, generative artificial intelligence (AI) has advanced rapidly, and localized large language models (LLM) are increasingly being built for lower-resource languages. Notable examples include Jais for Arabic (Sengupta et al., 2023), the SeaLLMs family for Southeast Asian languages (Zhang et al., 2025), and multilingual open releases such as Aya (Üstün et al., 2024). Since 2024, several Kazakh-adapted models have launched across multiple modalities: KazLLM (8B and 70B) (ISSAI, 2024) and Sherkala-8B (Koto et al., 2025) as LLMs; Qolda (Arystanbekov et al., 2026b) and HordeVision (Zubitskii et al., 2026) as vision-language models (VLM); and Qolda-AVL (Arystanbekov et al., 2026a) as an audio-language model (ALM). These releases are supported by Kazakh evaluation resources, including the open-domain QA dataset KazQAD (Yeshpanov et al., 2024), the multiple-choice benchmark KazMMLU (Togmanov et al., 2025), and a pilot

* Corresponding author.

evaluation of seven LLMs on Kazakh (Maxutov et al., 2024). General-purpose open models now also explicitly include Kazakh in pretraining. For example, Qwen3.5 supports 119 languages and dialects (Qwen Team, 2026), and Gemma 3 reports pretrained support for over 140 languages (Google DeepMind, 2026).

All these models, datasets, and benchmarks for Kazakh are built on Cyrillic text, which accounts for the large majority of written Kazakh. The national transition to the Latin alphabet raises a key question: how do LLMs serving Kazakh speakers, both local and global, perform when content is presented in Latin? This is important because the four proposed Latin alphabets differ in their treatment of Kazakh-specific phonemes. Previous studies on cross-script LLM performance, including the parallel Cyrillic and Latin entries for Kazakh and Uzbek in the recent TUMLU benchmark (Isbarov et al., 2025), have found accuracy drops related to script changes but have not identified the causes or tested solutions.

This question is relevant beyond Kazakh for three reasons. First, it provides a controlled setting to study how orthography affects LLMs, as only the alphabet changes while language and content remain constant. Second, the transitions described above show that Kazakh is not an isolated case, and romanization remains under consideration in other post-Soviet Turkic regions (Toktogulova et al., 2025). Quantitative evidence on LLM behavior during such transitions is therefore broadly valuable. Third, prior research on cross-lingual tokenizer fairness shows that higher tokenizer fertility, meaning more subword tokens per word, generally reduces performance on underrepresented languages (Rust et al., 2021; Ahia et al., 2023). However, for Kazakh, we observe the opposite: romanized text is more token-efficient than Cyrillic for most open models, yet accuracy on Latin still declines. Thus, the Cyrillic-to-Latin gap cannot be explained solely by tokenization costs.

In addition to these scientific motivations, this question is practically urgent. The Kazakh LLM, VLM, and ALM ecosystem described above was trained exclusively on Cyrillic text. If the transition to a Latin alphabet proceeds as planned, these models will require retraining, adaptation, or prompt-level mitigation to remain effective for Kazakh-speaking users. Determining which interventions are effective is therefore directly relevant to the country’s AI infrastructure during this transition.

In this work, we conduct a controlled diagnostic audit to establish an empirical baseline for the script transition, instead of proposing a new architectural approach. We evaluate 13 models: nine open instruction-tuned models with 2B to 10B parameters and four proprietary frontier models. Each is tested on three Kazakh benchmarks such as KazMMLU, KazCulture, and GSM8k-Kazakh. Every benchmark is presented in the original Cyrillic and in all four officially proposed Latin alphabets, converted with a deterministic, reproducible transliteration pipeline. We limit the evaluation to exact-match tasks because judge-model reliability for Latin Kazakh is not yet established. Exact-match scoring isolates the effect of orthographic variation without introducing confounds from an uncalibrated LLM judge. We measure tokenizer fertility for each model, dataset, and alphabet combination and find that, for most open models, the Cyrillic-Latin accuracy gap is not due to token-length inflation. We also test two prompt-level interventions: an explicit alphabet hint and a full Cyrillic-to-Latin character mapping in the prompt, reporting effects for each model. Finally, we apply targeted LoRA supervised fine-tuning on a small Latin/Cyrillic-mixed instruction dataset for three base models and two Latin variants, quantifying both recovery on the trained variant and Cyrillic forgetting. Our analysis is intentionally focused. We assess the most prominent mechanism in cross-lingual tokenization literature: subword fragmentation under script change, and report its empirical status. Two additional mechanisms, sparse coverage of Latin Kazakh subwords during pretraining and language-identification misrouting under Latin input, align with our observations but require separate experimental designs and will be addressed in future work. We therefore see this paper as an empirical baseline and a test of the dominant fragmentation hypothesis, rather than as a complete mechanistic account.

2 Related Work

2.1 Cross-alphabet behavior in language models

The TUMLU benchmark (Isbarov et al., 2025) is most similar to our work, evaluating LLMs on parallel Cyrillic and Latin entries for several Turkic languages, including Kazakh and Uzbek. While TUMLU identifies accuracy gaps between alphabets, it does not analyze their causes, test mitiga-

tions, or compare alternative Latin variants. For Uzbek, [Salaev et al. \(2022\)](#) presents a Cyrillic-Latin transliteration tool but does not examine its impact on LLM behavior. Previous research on transliteration and romanization in natural language processing (NLP) has mainly used alphabet conversion as a tool. It has been applied to improve cross-lingual transfer ([Amrhein and Sennrich, 2020](#)) and to enable multilingual capabilities in pretrained LLMs ([J et al., 2024](#)). The conversion itself has rarely been the object of evaluation. In contrast, our work holds language and content constant, varies only the alphabet, considers four suggested Latin variants for Kazakh, and reports both prompt-level and parameter-level interventions.

2.2 Tokenization fairness and low-resource adaptation

The tokenizer fertility framework of [Rust et al. \(2021\)](#) demonstrates that subword tokenizers optimized for high-resource languages generate longer sequences for underrepresented languages, which correlates with downstream performance loss. [Ahia et al. \(2023\)](#) extend this analysis to commercial application programmer interfaces (APIs), showing that the same content requires significantly more tokens and incurs higher costs in low-resource languages. We use the same fertility metric, tokens per whitespace-separated word, and apply it to each (model, dataset, alphabet) cell for Kazakh. This allows us to test whether the performance gap between Cyrillic and Latin scripts can be explained by tokenization cost. For adaptation, three main approaches are common for underserved languages: vocabulary expansion with continued pretraining, as in Jais ([Sengupta et al., 2023](#)) and the SeaLLMs family ([Zhang et al., 2025](#)); parameter-efficient fine-tuning using adapters such as LoRA ([Hu et al., 2022](#)); and instruction tuning on translated or natively collected data, as in Aya ([Üstün et al., 2024](#)). Our LoRA experiments belong to the second family, with one adapter trained per pair (base model, Latin variant).

3 Experimental Setup

3.1 Models

We evaluate 13 instruction-tuned models. Three are Kazakh-adapted: KazLLM-8B, Sherkala-8B, and Qolda (4.3B). To provide context, we include six general-purpose open-weight models with 2B to 10B parameters: Gemma-4-E2B-it, Gemma-4-

E4B-it, Qwen3.5-4B, Qwen3.5-9B, Llama-3.2-3B-Instruct, and Llama-3.1-8B-Instruct. To assess the impact of test-time reasoning on cross-alphabet performance, Qolda, Qwen3.5 and Gemma-4 are evaluated in both reasoning and non-reasoning modes. The other open models are run in their default non-thinking mode. Four proprietary frontier models, accessed via their official APIs, complete the set (GPT-4o, GPT-5, Gemini-2.5-Flash, Gemini-3.1-Flash; model identifiers, access dates, and decoding settings are given in Appendix C.1).

3.2 Benchmarks and Scoring

We evaluate three publicly available Kazakh benchmarks from Hugging Face, chosen to assess complementary skills and enable deterministic scoring. KazMMLU ([Togmanov et al., 2025](#)) measures subject-area knowledge using its Kazakh-language section, which includes 9,870 multiple-choice items across 12 high-school subjects. The Russian-language section is not considered in this study. KazCulture ([Maxutov et al., 2026](#)) assesses Kazakh cultural knowledge with 1,334 multiple-choice items from the test split. GSM8k-Kazakh, a Kazakh translation of GSM8k ([Cobbe et al., 2021](#)), contains 1,319 grade-school math word problems for mathematical reasoning and was introduced with the Qolda model ([Arystanbekov et al., 2026b](#)).

This selection is intentional, as all three benchmarks support exact-match scoring. In all three benchmarks, models are explicitly instructed in the prompt to return a JSON object containing the answer key. The value of this key is compared against the gold answer (a letter for KazMMLU and KazCulture, a numeric value for GSM8k-Kazakh). Prompt preambles and instruction templates for models were written in English in all cases to ensure a consistent instruction format across alphabets. The question content remains in Kazakh. Accuracy is calculated as

$$\text{Accuracy} = \frac{N_{\text{correct}}}{N_{\text{total}}} \quad (1)$$

per cell. We exclude open-ended generation benchmarks because scoring them would require an LLM-as-a-judge approach. Since judge model performance on Latin Kazakh is untested, any observed script effect could be confounded by judge unreliability under these conditions.

3.3 Alphabets and Transliteration

Each benchmark is evaluated using five alphabets: the original Cyrillic and four Latin variants proposed in 2017, 2018, 2019, and 2021¹. The corresponding Cyrillic-to-Latin mapping tables are provided in Appendix A. We use a deterministic character-level transliteration implemented in Python. All conversion scripts and evaluation code are publicly available.² For nine Cyrillic letters found in Russian loanwords in Kazakh text that lack equivalents in the four proposals (Ҫ, Ш, Ы, Э, Ю, Я, Ё, Ь, and Ъ), we introduce author-defined supplementary mappings to enable complete letter-to-letter conversion. These nine characters represent less than 0.5% of all text across the three benchmarks. The full frequency analysis, supplementary mapping decisions, and rationale are detailed in Appendix A. The 2019 variant also requires a context-sensitive rule for the vowel Ү, which depends on its position in the word and the vowel-harmony class of the surrounding context. We implement this according to the Kazakh Grammar convention (Mukhamedova, 2015).

Statistical noise floor. We analytically bound per-cell statistical uncertainty using only benchmark sizes. This approach provides a model-independent worst-case calibration that applies uniformly to all cells in Tables 1, 3, and 4. Therefore, treating each per-cell accuracy as a binomial proportion, the worst-case 95% Wald confidence half-width, attained at $p = 0.5$, is ± 2.70 points for GSM8k-Kazakh, ± 2.68 for KazCulture, and ± 0.99 for KazMMLU. For the unweighted three-benchmark averages, the corresponding worst-case half-width is ± 1.31 points. The full derivation, including the averaged-benchmarks bound, is given in Appendix E.

4 Cross-Alphabet Evaluation

We report baseline accuracy for each combination of model, benchmark, and alphabet across the 13 models. Table 1 summarizes these results, presenting per-model accuracy averaged over the three benchmarks for each alphabet, the Latin-only average, and the absolute drop Δ from Cyrillic to Latin

average. A detailed per-benchmark breakdown is provided in Appendix B.

All model configurations except GPT-5 experience significant accuracy loss when Cyrillic Kazakh is converted to Latin. The accuracy drop ranges from 1.4 points for GPT-5 to 30.4 points for Sherkala-8B. Excluding GPT-5, each model loses at least 5.6 points, more than twice the worst-case per-cell binomial bound on the smallest benchmark (Section 3), with most losing more than 14. GPT-5 is the only model whose 1.4 pp drop is within the averaged-benchmarks noise floor (± 1.31 pp; Appendix E), maintaining 86.9% accuracy on Latin compared with 88.3% on Cyrillic. No other model remains within 5 points of its Cyrillic baseline.

Two of the three Kazakh-adapted open-weight models, Sherkala-8B ($\Delta = 30.4$) and KazLLM-8B ($\Delta = 21.8$), show performance drops comparable to or exceeding those of non-adapted models of similar size. Qolda ($\Delta = 11.5$) is an exception, with a smaller drop than several non-adapted models of similar or larger size. This pattern suggests that Cyrillic-only adaptation, combined with Cyrillic-specific tokenizer extension, increases reliance on Cyrillic surface forms rather than developing script-agnostic Kazakh competence.

Of the four Latin proposals, the 2018 acute-accent variant is the easiest (47.6 overall), while the 2017 apostrophe-based variant is the most difficult (41.4). The 2019 and 2021 variants fall in between (45.5 to 45.9). The approximately 6-point difference among Latin variants is notable but much smaller than the roughly 17-point difference between Cyrillic and Latin scripts overall. In summary, the choice between Latin proposals is less significant than the decision to switch to a different script entirely.

For each model with a reasoning-mode variant (Gemma-4-E2B/E4B, Qwen3.5-4B/9B, Qolda), enabling reasoning increases both Cyrillic and Latin accuracy by 5 to 13 points, while keeping Δ within 1 to 5 points of the non-reasoning value. Reasoning does not selectively improve Latin; it raises both regimes equally. The script effect is additive to the reasoning effect and is not absorbed by it.

5 Tokenizer Fertility

A natural initial hypothesis is that the Latin drop results from the tokenization explosion. If a model’s BPE tokenizer has learned Cyrillic Kazakh subwords but not Latin Kazakh subwords, Latin in-

¹Our Latin renderings approximate the four official proposals as closely as transliteration permits and are not intended as canonical implementations. This study concerns the effect of script change on model behavior rather than the validity of any specific orthographic standard.

²https://github.com/IS2AI/Do-LLMs-cross-alphabets_code

Model	Cyrillic	Latin variant				Latin avg.	Δ
		2017	2018	2019	2021		
<i>Open-weight models</i>							
Llama 3.2-3B-Instruct	26.4	20.4	20.9	21.2	20.9	20.8	5.6
Llama 3.1-8B-Instruct	43.6	21.0	23.2	22.5	22.8	22.4	21.2
Gemma-4-E2B-it	48.5	23.3	26.5	24.3	24.8	24.7	23.8
Gemma-4-E2B-it (<i>think</i>)	56.6	26.9	32.3	28.8	29.2	29.3	27.3
Gemma-4-E4B-it	57.0	30.4	39.5	33.0	33.9	34.2	22.8
Gemma-4-E4B-it (<i>think</i>)	65.8	38.1	51.1	44.8	44.6	44.6	21.2
Qwen3.5-4B	57.1	33.2	40.9	37.4	37.9	37.4	19.7
Qwen3.5-4B (<i>think</i>)	66.7	42.6	54.9	50.3	49.4	49.3	17.4
Qwen3.5-9B	67.3	49.2	55.2	53.3	54.8	53.1	14.2
Qwen3.5-9B (<i>think</i>)	71.5	49.6	62.7	58.7	58.5	57.4	14.1
<i>Kazakh-adapted models</i>							
KazLLM-8B	47.4	23.1	27.4	24.6	27.2	25.6	21.8
Sherkala-8B	57.2	24.7	27.2	26.9	28.5	26.8	30.4
Qolda	60.4	45.3	48.9	49.8	51.8	48.9	11.5
Qolda (<i>think</i>)	61.3	44.7	48.3	49.7	48.5	47.8	13.5
<i>Proprietary models</i>							
Gemini-2.5-Flash	75.5	58.3	67.5	64.0	63.9	63.4	12.1
Gemini-3.1-Flash	84.5	72.1	78.2	76.8	76.6	75.9	8.6
GPT-4o	78.5	56.1	65.3	65.9	66.5	63.5	15.0
GPT-5	88.3	86.0	87.0	87.4	87.1	86.9	1.4

Table 1: Baseline accuracy (%) is averaged across KazMMLU, KazCulture, and GSM8k-Kazakh. *Latin avg.* represents the mean of the four Latin variants. Δ indicates the absolute decrease from *Cyrillic* to *Latin avg.* in percentage points. (*think*) refers to reasoning-mode runs of the same model.

put would fragment into more tokens, exhausting the context budget and reducing performance. We test this prediction by measuring tokenizer fertility across all model, benchmark, and alphabet combinations.

We measure fertility, defined as the average number of tokens produced per whitespace-separated word, for each of the nine open-weight models with accessible tokenizers across all three benchmarks and five alphabets. Table 2 presents the average fertility across the benchmarks and the change Δ_{fert} from Cyrillic to Latin Average.

For most tokenizers in our suite, including Llama, KazLLM (which uses the Llama tokenizer), Qolda, Qwen3.5, and Gemma-4, Latin Kazakh requires the same number of tokens per word or fewer than Cyrillic Kazakh. Llama and KazLLM both decrease by 1.13 tokens per word on Latin, while Qolda drops by 0.99. Qwen3.5 and Gemma-4 show minimal change. However, these models lose 5 to 28 accuracy points on Latin (Table 1). Tokenization alone does not explain the magnitude of this difference.

Sherkala-8B is the only model whose fertility increases on Latin (*Cyrillic* 2.22 to *Latin avg.* 3.09, +0.87). During adaptation, Sherkala’s tokenizer was extended with Kazakh-Cyrillic subword merges, resulting in exceptionally low Cyrillic fer-

tility (Koto et al., 2025). Switching to Latin restores fertility to levels comparable to those of other tokenizers. Sherkala also exhibits the largest accuracy drop in our study ($\Delta = 30.4$). The alignment between fertility and accuracy supports the prediction of the hypothesis, but this pattern is observed only in this model.

Pooling all nine tokenizers, the Pearson correlation between Δ_{fert} and accuracy drop Δ is $r = +0.36$ in the direction predicted by the hypothesis but weak-to-moderate (see Appendix D). Excluding Sherkala-8B weakens the association to $r = +0.26$ but does not eliminate it. We conclude that although fertility correlates with the drop as predicted, its effect is too minor to explain the observed accuracy gap. Tokenization contributes but is not the primary factor. The remaining gap likely results from two factors identified in Section 1: limited coverage of Latin Kazakh subwords during pretraining and language-identification errors with Latin input. Empirical isolation of these factors requires separate experiments and will be addressed in future work.

6 Mitigations

If the script effect is not caused by tokenization (Section 5), can it be addressed at inference time

Tokenizer (model)	Cyrillic	Latin variant				Latin avg.	Δ_{fert}
		2017	2018	2019	2021		
Gemma-4-E2B-it / E4B-it	3.38	3.87	3.39	3.38	3.31	3.48	+0.11
Qwen3.5-4B / 9B	3.59	3.69	3.56	3.52	3.37	3.53	-0.06
Llama 3.1/3.2-Instruct	4.64	3.65	3.55	3.48	3.36	3.51	-1.13
KazLLM-8B	4.64	3.65	3.55	3.48	3.36	3.51	-1.13
Qolda	4.62	3.82	3.62	3.58	3.52	3.63	-0.99
Sherkala-8B	2.22	3.46	2.71	3.02	3.17	3.09	+0.87

Table 2: Tokenizer fertility (tokens per word), averaged across KazMMLU, KazCulture, and GSM8k-Kazakh. Δ_{fert} is the change from *Cyrillic* to *Latin avg.* in tokens per word; a positive value means Latin requires more tokens and vice versa. Tokenizers shared by multiple models in our suite are reported once.

through prompting, or does it require updating model parameters? We evaluate both approaches, starting with the least invasive.

6.1 Prompt-Level Mitigations

We test two prompt-level interventions, applying both only to the Latin-alphabet cells while leaving the Cyrillic baseline unchanged:

Hint. The prompt begins with a one-line statement indicating that the Kazakh text uses the Latin alphabet (e.g., “*The following Kazakh text uses the Latin alphabet.*”). This approach tests whether explicit language and script identification alone can restore Latin performance.

Mapping. The prompt begins with the complete Cyrillic-to-Latin letter table for the active Latin variant (Appendix A), followed by the baseline question template. This setup tests whether explicit decoding information enables the model to map Latin input back to its Cyrillic representation.

Table 3 reports the Latin Average accuracy under each condition for the fourteen open-weight model configurations. We limited the mitigation study to open-weight models to enable a consistent comparison of prompt-level and parameter-level interventions. Open models support both types of interventions, while proprietary endpoints support only prompting. For GPT-5, the baseline drop of 1.4 points is already near our averaged-benchmarks uncertainty bound (Appendix E), making a mitigation study less informative than for models with larger performance gaps.

On average across the nine models, both hint and mapping increase scores by about +1 point over the Latin baseline, within the averaged-benchmarks noise floor of ± 1.31 points (Section 3), and only a small portion of the approximately 17-point decrease from Cyrillic. The highest individual improvement is +6.7 points (Qolda *think*, under mapping), while six configurations show a negative

change under at least one intervention. Importantly, no intervention restores performance to within 5 points of the Cyrillic baseline. On this evidence, the two simple interventions tested here do not appear sufficient on their own.

6.2 Parameter-Level Mitigation: LoRA Fine-Tuning

We evaluate whether brief parameter updates can restore Latin performance. We train low-rank adapters (Hu et al., 2022) for each of Qwen3.5-4B, KazLLM-8B, and Llama-3.2-3B-Instruct, separately for the 2017 and 2021 Latin variants, resulting in eight adapters. Training data is drawn from the KazCulture training split, which consists of *item description + question–answer* pairs. To mitigate Cyrillic forgetting, 85% of training examples are transliterated to the target Latin variant and the remaining 15% are retained in Cyrillic; full hyperparameters are provided in Appendix C. Each adapter is evaluated on both Latin and Cyrillic to measure recovery and potential Cyrillic-forgetting. Table 4 presents accuracy averaged across the three benchmarks.

The two adapters trained on Qwen3.5-4B in thinking mode close most of the Latin gap, improving scores by +20.0 points on the 2017 variant and +13.2 on the 2021, with negligible impact on Cyrillic performance (-0.3 and +0.1 respectively). These adapters raise Latin accuracy to 62–63%, within 4 points of the unadapted Cyrillic baseline. This represents the largest recovery achieved by any mitigation in this study.

Applying the same LoRA recipe to Qwen3.5-4B without thinking mode recovers only +4 to +5 points on Latin but loses 3.5 to 8.3 points on Cyrillic, which is not a favorable trade-off. For KazLLM-8B, a model already fine-tuned for Kazakh Cyrillic, the outcome is worse: Latin improves only slightly (+0.5 to +2.1), while Cyril-

Model	Cyrillic base	Latin base	Latin hint	Latin map	Δ_{best}
Llama 3.2-3B-Instruct	26.4	20.8	20.7	20.0	-0.2
Llama 3.1-8B-Instruct	43.6	22.4	18.2	22.6	+0.3
Gemma-4-E2B-it	48.5	24.7	26.3	27.2	+2.5
Gemma-4-E2B-it (<i>think</i>)	56.6	29.3	29.8	30.4	+1.1
Gemma-4-E4B-it	57.0	34.2	40.2	38.0	+6.0
Gemma-4-E4B-it (<i>think</i>)	65.8	44.6	48.4	46.1	+3.8
Qwen3.5-4B	57.1	37.4	41.9	41.8	+4.5
Qwen3.5-4B (<i>think</i>)	66.7	49.3	46.5	46.6	-2.7
Qwen3.5-9B	67.3	53.1	54.9	54.1	+1.8
Qwen3.5-9B (<i>think</i>)	71.5	57.4	56.9	54.6	-0.5
KazLLM-8B	47.4	25.6	24.6	24.2	-1.0
Sherkala-8B	57.2	26.8	27.0	26.1	+0.1
Qolda	60.4	48.9	49.7	49.3	+0.8
Qolda (<i>think</i>)	61.3	47.8	53.2	54.5	+6.7

Table 3: *Latin base / hint / map* represent mean accuracy (%) across four Latin variants and three benchmarks for the baseline, hint, and mapping conditions. Δ_{best} is the greater of (hint - baseline, mapping - baseline).

Base	Mode	Var	Cyrillic			Latin (variant)		
			Base	SFT	Δ	Base	SFT	Δ
Qwen3.5-4B	no-think	2017	56.8	48.5	-8.3	33.6	38.7	+5.1
Qwen3.5-4B	no-think	2021	56.8	53.3	-3.5	38.9	42.9	+4.0
Qwen3.5-4B	think	2017	66.7	66.5	-0.3	42.6	62.6	+20.0
Qwen3.5-4B	think	2021	66.7	66.9	+0.1	49.7	62.9	+13.2
KazLLM-8B	—	2017	47.4	37.9	-9.5	23.1	25.1	+2.1
KazLLM-8B	—	2021	47.4	41.3	-6.1	27.2	27.7	+0.5
Llama-3.2-3B-Instruct	—	2017	26.4	20.2	-6.2	20.4	20.6	+0.2
Llama-3.2-3B-Instruct	—	2021	26.4	22.2	-4.2	20.9	21.5	+0.6

Table 4: LoRA fine-tuning results. *Cyrillic* columns measure forgetting; *Latin* columns measure recovery on the targeted variant. Both blocks report accuracy (%) averaged over KazMMLU, KazCulture, and GSM8k-Kazakh. Each row is one adapter, trained per (base, mode, variant).

lic performance drops significantly (-6.1 to -9.5). Llama-3.2-3B-Instruct shows a similar trend, with Latin scores remaining stable and Cyrillic scores declining. The Kazakh adaptation, which previously reduced KazLLM’s performance in cross-alphabet evaluation (Section 4), now becomes a liability under LoRA. The Cyrillic representation is fragile enough to be disrupted by minor adapter updates that do not improve Latin competence.

The best result in the study is achieved by Qwen3.5-4B thinking mode with a per-variant Latin adapter. This approach combines two effects that are effective individually: thinking mode offers chain-of-thought scaffolding that allows the Latin input to be re-anchored dynamically, while the LoRA adapter enables the model to interpret Latin characters. Neither method alone recovers more than 10 points, but together they recover between 13 and 20 points.

7 Discussion

7.1 What Helps Close the Gap

The two simple prompt-level interventions we tested (a one-line script hint and a verbatim character mapping table) are cost-effective but yield only modest and inconsistent improvements. The average gain is approximately +1 point across fourteen configurations, and some models experience reduced performance with hint or mapping. Parameter-level interventions can close most of the gap, achieving up to +20 points on Latin with minimal Cyrillic forgetting in the best case (Qwen3.5-4B *thinking* + LoRA). However, this is possible only if the base model is large enough to benefit from adaptation and is not already highly specialized for Cyrillic. KazLLM-8B and Llama-3.2-3B do not meet these criteria: KazLLM is too committed to Cyrillic to adapt without forgetting, while Llama-3.2-3B is too small to learn the new representation effectively.

A natural system-level workaround for the Cyrillic–Latin gap is reverse transliteration: prepro-

cessing Latin user input back into Cyrillic before passing it to a Cyrillic-trained model. This is a reasonable deployment wrapper, and we would expect it to recover a substantial part of the gap in practice. It also addresses a somewhat different question from the one studied here, namely whether models can be wrapped to cope with Latin input rather than whether they process it natively. We note that the inverse mapping is not fully deterministic, since several proposals merge distinct Cyrillic letters: for example, И and I both map to I in the 2017 variant, and X and Һ map to H in all four proposals. The context-sensitive treatment of Y in the 2019 variant would likewise have to be inverted with reference to the surrounding context. Quantifying the resulting round-trip loss seems a useful direction for deployment-oriented work.

7.2 Why Models May Differ

Among open-weight models, Sherkala-8B ($\Delta = 30.4$) and KazLLM-8B ($\Delta = 21.8$) show the largest accuracy loss on Latin, both having the most explicit Kazakh adaptation. This suggests that fine-tuning on Cyrillic-only Kazakh reduces generalization, as improvements on Cyrillic Kazakh may rely on Cyrillic-specific subword patterns rather than script-agnostic Kazakh skills. Qolda, which maintains better Latin performance ($\Delta = 11.5$), shows that this limitation is not unavoidable. Developing Kazakh-adapted models that transfer well to Latin requires attention to script diversity during adaptation.

Of the three Kazakh-adapted open-weight models, only Qolda ($\Delta = 11.5$) drops less than non-adapted peers of similar or larger size (Llama-3.1-8B 21.2, Gemma-4-E2B-it 23.8, GPT-4o 15.0). Three properties distinguish it from Sherkala-8B and KazLLM-8B: (i) it is a vision-language model adapted on image-text pairs alongside Kazakh text, broadening its text exposure beyond Kazakh-Cyrillic; (ii) its tokenizer was not extended with Kazakh-Cyrillic merges, so its Cyrillic fertility (4.62, Table 2) matches the Llama family rather than the heavily extended Sherkala-8B (2.22), which posts the largest Latin drop in our suite; (iii) it is smaller (4.3B vs. 8B). Reasoning capability is a fourth nominal difference but does not reduce Qolda’s gap at inference time (Qolda *think* $\Delta = 13.5$ vs. Qolda $\Delta = 11.5$); reasoning interacts with parameter-level adaptation rather than baseline cross-script transfer. Taken together, this suggests that poor cross-script transfer is linked

most closely to Cyrillic-specific tokenizer extensions and Cyrillic-only data mixes, not to Kazakh adaptation as such.

GPT-5 is the only model whose 1.4 pp drop falls within our averaged-benchmarks noise floor (± 1.31 pp; Appendix E). With no access to its training data or tokenizer, any account is speculative. Three non-exclusive possibilities are consistent with the observation. The first is substantially more Latin Kazakh in pretraining than is visible on the open web, such as diaspora content, parallel transliteration resources, and official documentation from the four Latin proposals. The second is implicit normalization of Latin input to a Cyrillic-like internal representation, and the third is language identification that is robust to script change. Distinguishing these requires inspection of reasoning traces, which we leave to future work. The practical implication is bounded: a single robust proprietary model does not substitute for transition-ready open infrastructure, which serves the population most exposed to the change.

7.3 Implications

The planned Cyrillic-to-Latin transition is likely to present significant challenges for current Kazakh NLP infrastructure. Models trained on Cyrillic Kazakh, even those specifically designed for it, do not generalize to any of the four proposed Latin variants. Prompt-level patches are inadequate, and parameter-level patches are only effective for certain base-model configurations, often resulting in a loss of Cyrillic performance. To address this, future Kazakh language models should be pretrained on a mix of Latin and Cyrillic data from the outset, rather than relying on retrofitted adapters.

Of the four Latin proposals, the 2018 acute-accent variant is the easiest for current models (47.6 overall), while the 2017 apostrophe-based variant is the most challenging (41.4). The 2019 and 2021 variants fall in between. Although there is a 6-point difference among Latin variants, this is much smaller than the 17-point gap between Cyrillic and Latin scripts overall. For policymakers, this suggests that the decision to change scripts is more significant than the choice among Latin variants. Investments in transition-ready training data will be valuable regardless of the variant selected.

The quantitative effect sizes reported here are specific to Kazakh and should not be carried over to other languages without direct evaluation. The audit design itself may be more broadly applica-

ble: holding language and content constant while varying only the orthography offers a simple way of probing model robustness under script change. Uzbek, for instance, shows cross-script differences between parallel Cyrillic and Latin entries in the TUMLU benchmark (Isbarov et al., 2025), and script reform remains under discussion for Kyrgyz (Toktogulova et al., 2025). A comparable audit would likely be informative in those settings as well.

8 Conclusion

We assessed how current LLMs handle the Kazakh script change from Cyrillic to Latin using four official Latin proposals, three benchmarks, and 13 models. On average, accuracy drops by 16.8 points for Latin, a trend seen across nearly all model families except GPT-5. This decline is not primarily explained by tokenizer fertility, which remains stable or decreases for most tokenizers; tokenization contributes, but its effect is insufficient to account for the observed magnitude. Kazakh-adapted open-weight models experience the largest drop, suggesting that Cyrillic-only adaptation increases script dependence rather than fostering script-agnostic Kazakh proficiency. The two prompt-level interventions we tested yield only minor improvements, while per-variant LoRA applied to strong reasoning-capable models restores most lost performance. Three findings support this: the scale and consistency of the cross-script drop, the failure of subword fragmentation to explain it, and the limited effect of inference-time patches. Together they suggest that the script gap cannot be repaired after the transition is under way. Effective preparation for Kazakh NLP requires Latin Kazakh data during pretraining.

9 Limitations

We tested subword fragmentation as a potential mechanism (Section 5) and determined it is not the primary cause. Two additional candidates, sparse Latin-Kazakh coverage in pretraining and language-identification routing under Latin input, align with our observations but require separate investigations. These include probing language identification under Latin input or measuring vocabulary coverage using held-out Latin Kazakh corpora, which we leave for future work. Our mitigation conclusions (Section 6) remain valid regardless of which factor is more significant. The

Latin benchmarks are generated through deterministic character-level transliteration, isolating the script variable but not addressing naturalistic Latin Kazakh.

Our prompting and adaptation results are limited in scope. We tested two basic prompt-level interventions: a one-line script hint and a verbatim mapping table. We did not examine few-shot Latin-Cyrillic exemplars, explicit transliteration chains, self-consistency voting, or preambles written in Kazakh. Since all preambles were in English (Section 3.2), the effect of prompt language remains untested. Therefore, our conclusions apply only to the simple interventions evaluated. Similarly, the LoRA experiments used three base models, two Latin variants, and a fixed 85/15 Latin-to-Cyrillic mix, without varying the ratio, adapter rank, learning rate, or training domain. As a result, the recovery-forgetting trade-off curve is not mapped, and the reported results represent a single, non-optimized point.

We use exact match scoring on JSON-extracted answers. Open-ended generation, dialogue, and summarization in Latin Kazakh were not evaluated because reliable LLM-as-a-judge protocols for this language are not yet established and require calibration with native-speaker labels. We bound per-cell uncertainty analytically (Appendix E) instead of using per-cell bootstrap. Per-item predictions are included with the code release.

Ethics Statement

This study evaluates and diagnoses large language models in relation to the planned Kazakh script transition from Cyrillic to Latin. We use all available models and benchmarks according to their intended research use: open models are accessed via public weights, while proprietary models are used through official APIs under the providers’ terms of service. We release deterministic Cyrillic-to-Latin conversion scripts and evaluation code to support reproducibility and further work on the script transition. We do not release new pretrained models. The LoRA adapters we trained serve as research artifacts, demonstrating that targeted recovery is feasible only under specific base-model conditions, and are not intended for production use.

This paper addresses a societal risk that differs from the usual concern: the evaluated models may lack sufficient capability to support Kazakh speakers during the official script transition. Section 7

provides a detailed discussion. We do not identify significant dual-use risks in releasing deterministic transliteration scripts, as the four Latin proposals are public-domain orthographic standards published by the Government of the Republic of Kazakhstan. Our Latin-script benchmarks are based on publicly available Cyrillic benchmarks and do not increase the knowledge base or attack surface of any model.

This study uses three benchmarks: multiple-choice high school knowledge questions (KazMMLU), Kazakh cultural knowledge questions (KazCulture), and grade school math word problems (GSM8k-Kazakh). These benchmarks do not include personally identifying information or offensive content, as confirmed during the creation of the Latin transliterations. Our deterministic, character-level transliteration pipeline preserves all content, identities, and references.

All artifacts used in this study are publicly available under their original licenses. Our use is strictly for non-commercial research purposes. Open-weight models (Llama 3.1/3.2, Gemma-4, Qwen3.5, KazLLM-8B, Sherkala-8B, Qolda) are used under the research or community licenses specified on their HuggingFace model cards. Proprietary models (GPT-4o, GPT-5, Gemini-2.5/3.1-Flash) are accessed through official APIs in accordance with the providers' terms of service. Benchmarks (KazMMLU, KazCulture, GSM8k-Kazakh) are used under the licenses specified in their source papers, all of which permit research use. Our conversion scripts and evaluation code are publicly available.

Acknowledgements

This research is funded by the Committee of Science of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. BR24993001). The authors would like to thank Madina Mansurova for supervising the grant that facilitated this research.

References

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Junjo Kasai, David Mortensen, Noah Smith, and Yulia Tsvetkov. 2023. [Do all languages cost the same? tokenization in the era of commercial language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9904–9923, Singapore. Association for Computational Linguistics.

Chantal Amrhein and Rico Sennrich. 2020. [On Romanization for model transfer between scripts in neural machine translation](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2461–2469, Online. Association for Computational Linguistics.

Batyr Arystanbekov, Akylbek Maxutov, Aspandiyar Nurimanov, and Huseyin Atakan Varol. 2026a. [Extending a vision–language model with audio understanding: Introducing Qolda-AVL for the kazakh language](#). *Big Data and Cognitive Computing*, 10(6).

Batyr Arystanbekov, Aspandiyar Nurimanov, Akylbek Maxutov, Vladimir Albrekht, Askat Kuzdeuov, and Huseyin Atakan Varol. 2026b. [Qolda: A small vision–language model for the Kazakh language](#). *IEEE Access*, 14:46392–46414.

Zhazira Bekzhanova and Tsediso Michael Makoele. 2022. [Latinization of the Kazakh alphabet: Implications for education, inclusion, and social cohesion in Kazakhstan](#). *SAGE Open*, 12(4).

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.

Google DeepMind. 2026. Gemma 4: A gemma collection on hugging face. <https://huggingface.co/collections/google/gemma-4>.

Lynley Hatcher. 2008. [Script change in Azerbaijan: acts of identity](#). *International Journal of the Sociology of Language*, 2008(192):105–116.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.

Jafar Isbarov, Arofat Akhundjanova, Mammad Hajili, Kavsar Huseynova, Dmitry Gaynullin, Anar Rzayev, Osman Tursun, Aizirek Turdubaeva, Ilshat Saetov, Rinat Kharisov, Saule Belginova, Ariana Kenbayeva, Amina Alisheva, Abdullatif Köksal, Samir Rustamov, and Duygu Ataman. 2025. [TUMLU: A unified and native language understanding benchmark for Turkic languages](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22816–22838, Vienna, Austria. Association for Computational Linguistics.

ISSAI. 2024. KazLLM: An open generative large language model for Kazakh. <https://huggingface.co/issai/LLama-3.1-KazLLM-1.0-8B>.

Jaavid J, Raj Dabre, Aswanth M, Jay Gala, Thanmay Jayakumar, Ratish Puduppully, and Anoop Kunchukuttan. 2024. [RomanSetu: Efficiently unlocking multilingual capabilities of large language](#)

- models via Romanization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15593–15615, Bangkok, Thailand. Association for Computational Linguistics.
- Fajri Koto, Rituraj Joshi, Nurdaulet Mukhituly, Yuxia Wang, Zhuohan Xie, Rahul Pal, Daniil Orel, Parvez Mullah, Diana Turmakhan, Maiya Goloburda, Mohammed Kamran, Samujjwal Ghosh, Bokang Jia, Jonibek Mansurov, Mukhammed Togmanov, Debopriyo Banerjee, Nurkhan Laiyk, Akhmed Sakip, Xudong Han, and 15 others. 2025. *Sherkala-chat: Building a state-of-the-art LLM for kazakh in a moderately resourced setting*. In *Second Conference on Language Modeling*.
- Geoffrey Lewis. 1999. *The Turkish Language Reform: A Catastrophic Success*. Oxford University Press.
- Akylbek Maxutov, Batyr Arystanbekov, Zhanat Makhataeva, Adil Yergen, Nurbek Taizhanov, Guldana Nauryzbaikyzy, and Huseyin Atakan Varol. 2026. *Introducing cultural knowledge in language models: KazCulture dataset for Kazakh culture*. *IEEE Access*, 14:44027–44042.
- Akylbek Maxutov, Ayan Myrzakhmet, and Pavel Braslavski. 2024. *Do LLMs speak Kazakh? a pilot evaluation of seven models*. In *Proceedings of the First Workshop on Natural Language Processing for Turkic Languages (SIGTURK 2024)*, pages 81–91, Bangkok, Thailand and Online. Association for Computational Linguistics.
- Raikhangul Mukhamedova. 2015. *Kazakh: A Comprehensive Grammar*. Routledge Comprehensive Grammars. Routledge, London.
- Qwen Team. 2026. Qwen3.5: A qwen collection on hugging face. <https://huggingface.co/collections/Qwen/qwen35>.
- Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. *How good is your tokenizer? On the monolingual performance of multilingual language models*. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.
- Ulugbek Salaev, Elmurod Kuriyozov, and Carlos Gómez-Rodríguez. 2022. *A machine transliteration tool between Uzbek alphabets*. In *Proceedings of the ALT/NLP The International Conference and workshop on Agglutinative Language Technologies as a challenge of Natural Language Processing*, volume 3315 of *CEUR Workshop Proceedings*, pages 42–50. CEUR-WS.org.
- Neha Sengupta, Sunil Kumar Sahu, Bokang Jia, Satheesh Katipomu, Haonan Li, Fajri Koto, William Marshall, Gurpreet Gosal, Cynthia Liu, Zhiming Chen, Osama Mohammed Afzal, Samta Kamboj, Onkar Pandit, Rahul Pal, Lalit Pradhan, Zain Muhammad Mujahid, Massa Baali, Xudong Han, Sondos Mahmoud Bsharat, and 13 others. 2023. *Jais and jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models*. *Preprint*, arXiv:2308.16149.
- Mukhammed Togmanov, Nurdaulet Mukhituly, Diana Turmakhan, Jonibek Mansurov, Maiya Goloburda, Akhmed Sakip, Zhuohan Xie, Yuxia Wang, Bekassyl Syzdykov, Nurkhan Laiyk, Alham Fikri Aji, Ekaterina Kochmar, Preslav Nakov, and Fajri Koto. 2025. *KazMMLU: Evaluating language models on Kazakh, Russian, and regional knowledge of Kazakhstan*. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14403–14416, Vienna, Austria. Association for Computational Linguistics.
- Gulzhamal Toktogulova, Sayipzhamal Sakieva, Sabira Osmonalieva, Kubanychbek Uraimov, and Meergul Asanalieva. 2025. *Prospects of Kyrgyz language transition to the Latin script based on experience of Kazakhstan and Uzbekistan*. *Language Problems and Language Planning*, 49(2):129–152.
- Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D’souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Fredie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. 2024. *Aya model: An instruction fine-tuned open-access multilingual language model*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15894–15939, Bangkok, Thailand. Association for Computational Linguistics.
- Mehmet Uzman. 2010. *Romanisation in Uzbekistan past and present*. *Journal of the Royal Asiatic Society*, 20(1):49–60.
- Rustem Yeshpanov, Pavel Efimov, Leonid Boytsov, Ardak Shalkarbayuli, and Pavel Braslavski. 2024. *KazQAD: Kazakh open-domain question answering dataset*. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9645–9656, Torino, Italia. ELRA and ICCL.
- Wenxuan Zhang, Hou Pong Chan, Yiran Zhao, Mahani Aljunied, Jianyu Wang, Chaoqun Liu, Yue Deng, Zhiqiang Hu, Weiwen Xu, Yew Ken Chia, Xin Li, and Lidong Bing. 2025. *SeaLLMs 3: Open foundation and chat multilingual large language models for Southeast Asian languages*. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 96–105, Albuquerque, New Mexico. Association for Computational Linguistics.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyang Luo. 2024. [LlamaFactory: Unified efficient fine-tuning of 100+ language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

Pavel Zubitskii, Vitaliy Morozov, Sanzhar Murzakmetov, Beksultan Sagyndyk, and Sanzhar Umbet. 2026. [HordeVision: An open-source Kazakh vision-language model](#). *TechRxiv*, 2026(0110).

A Transliteration Details

A.1 Example Sentence Across Scripts

To illustrate script differences, we present the same Kazakh sentence in all five evaluated scripts, along with its English translation, transliterated using the deterministic pipeline described in Section 3. The sentence is relevant to this paper and includes several Kazakh-specific letters (Ә, Ғ, Ө, Ұ, Қ, І) to highlight orthographic variations.

Script	Sentence
English	<i>The transition to the new alphabet will be completed in 2031.</i>
Cyrillic	Жаңа әліпбиге көшу 2031 жылы аяқталады.
Latin 2017	Jan’a a’lipbige ko’shy’ 2031 jyly ayaqtalady.
Latin 2018	Jaña älipbige kóshý 2031 jyly ayaqtalady.
Latin 2019	Jaña älipbige köşw 2031 jyly ayaqtalady.
Latin 2021	Jaña älipbige köşu 2031 jyly ayaqtalady.

Table 5: Example Kazakh sentence in all five evaluated scripts. The Latin transliterations were generated by the deterministic pipeline of Section 3 and should be interpreted as experimental renderings rather than canonical orthographic realizations.

A.2 Unified Cyrillic-to-Latin Conversion Table

Table 7 gives the complete letter-by-letter mapping used for each of the four proposed Latin variants. Mappings for nine Cyrillic letters that appear in Russian loanwords but are not covered by any proposal (Ё, Ъ, Ч, Ш, Ъ, Ь, Э, Ю, Я) are author-defined supplementary mappings, with rationale in Section A.4.

A.3 Frequency of Unrepresented Cyrillic Letters

Table 6 reports the frequency of the nine unrepresented Cyrillic letters as a percentage of total

characters in each benchmark. The combined frequency of these letters never exceeds 0.5% in any benchmark, justifying author-defined supplementary mappings rather than a heavier intervention such as filtering or dual-stream transliteration.

Cyrillic	KazCulture	GSM8k-Kazakh	KazMMLU
Ё	0.0000%	0.0000%	0.0003%
Ю	0.0427%	0.0719%	0.0292%
Я	0.1693%	0.1988%	0.2677%
Ъ	0.0001%	0.0000%	0.0020%
Ь	0.0112%	0.1475%	0.0680%
Ч	0.0032%	0.0681%	0.0145%
Ш	0.0019%	0.0000%	0.0029%
Ц	0.0101%	0.0842%	0.1079%
Э	0.0178%	0.0998%	0.0872%

Table 6: Frequency of unrepresented Cyrillic letters as a percentage of total characters per benchmark. Corpus sizes: KazCulture (9,730,536 chars), GSM8k-Kazakh (684,263), KazMMLU (2,393,729).

A.4 Rationale for Supplementary Mappings

The nine Cyrillic letters without Latin equivalents in any of the four proposals are addressed as follows. Ч is rendered as ch, or as ç in the 2019 variant, following standard Slavic/Turkic romanization and Turkish orthographic conventions. Ш is rendered as sch in all variants, in line with Central European convention. Ц is rendered as ts by default, but as s in the digraph CIЦ to avoid the cluster sts. Э is rendered as e in all variants, as it represents the same front unrounded vowel as E in Kazakh phonology. Ю, Я, and Ё are decomposed into their constituent phonemes (й + y/a/o), with the vowel prefix determined by each variant’s treatment of И. Ъ and Ь are omitted in all variants, as they do not carry independent phonemic value in Kazakh.

B Per-Benchmark Cross-Alphabet Results

Tables 8, 9 and 10 present baseline, hint, and mapping accuracy for each benchmark. The overall trends from Section 4 are consistent across benchmarks: a near-universal drop, GPT-5 immunity, and Sherkala-8B experiencing the largest single drop (KazCulture, $\Delta = 27.7$). Math reasoning (GSM8k-Kazakh) exhibits the most significant baseline declines for small and general-purpose models (Gemma-4-E2B-it $\Delta > 54$, Llama 3.1-8B $\Delta = 39.8$), indicating that script changes particularly affect multi-step arithmetic reasoning. The

#	Cyr	2017	2018	2019	2021	#	Cyr	2017	2018	2019	2021
1	А	A	A	A	A	22	П	P	P	P	P
2	Ә	A'	Á	Ä	Ä	23	Р	R	R	R	R
3	Б	B	B	B	B	24	С	S	S	S	S
4	В	V	V	V	V	25	Т	T	T	T	T
5	Г	G	G	G	G	26	У	Y'	Ý	W ^a	U
6	Ғ	G'	Ġ	Ğ	Ğ	27	Ұ	U	U	U	Ü
7	Д	D	D	D	D	28	Ү	U'	Ú	Ü	Ü
8	Е	E	E	E	E	29	Ф	F	F	F	F
9	Ё	i'o	io	io	io	30	Х	H	H	H	H
10	Ж	J	J	J	J	31	Һ	H	H	H	H
11	З	Z	Z	Z	Z	32	Ц	Ts	Ts	Ts	Ts
12	И	I	I	İ	İ	33	Ч	C'	Ch	Ç	Ch
13	Й	I'	ı	ı	İ	34	Ш	S'	Sh	Ş	Ş
14	К	K	K	K	K	35	Щ	Sch	Sch	Sch	Sch
15	Қ	Q	Q	Q	Q	36	Ъ	—	—	—	—
16	Л	L	L	L	L	37	Ы	Y	Y	Y	Y
17	М	M	M	M	M	38	І	I	—	—	ı
18	Н	N	N	N	N	39	Ь	—	—	—	—
19	Ң	N'	Ñ	Ñ	Ñ	40	Ә	E	E	E	E
20	О	O	O	O	O	41	Ю	i'y	ıy	ıu	ıu
21	Ө	O'	Ó	Ö	Ö	42	Я	i'a	ia	ia	ia

^a In the 2019 mapping, the representation of Y is context-dependent: it is W when adjacent to a vowel, and U, Ü, or Ü elsewhere based on vowel harmony rules.

Table 7: Unified Cyrillic-to-Latin conversion table covering all four proposed Kazakh Latin alphabets, presented in two side-by-side halves for clarity. The symbol “—” indicates that the letter is omitted by the relevant source decree. Dotted and dotless I are represented in different cases for clarity.

minimal average impact of hint and mapping noted in Section 6.1 is also observed within each benchmark.

C Model and Training Configuration Details

C.1 Proprietary Model API Configuration

Table 11 summarises the API configuration used for the four proprietary models, all of which were queried through their official developer APIs. Decoding settings are not identical across providers. GPT-4o was queried with temperature 0.0, whereas for GPT-5 no temperature was sent and the API default applies; for the two Gemini models no generation configuration was sent at all, so temperature, top-p, maximum output length and thinking behaviour follow the service defaults. No explicit reasoning-effort parameter was set for GPT-5. For Gemini, a request that failed after five retries returned an empty response, which we counted as a failed answer extraction. These settings were fixed before evaluation and applied identically across all alphabets and benchmarks, so any residual sampling variation affects the Cyrillic and Latin conditions equally.

C.2 LoRA Training Hyperparameters

Table 12 reports the hyperparameters used for all eight LoRA adapters trained in Section 6.2. The same configuration is applied to every (base model, mode, Latin variant) combination; differences in outcome are attributable to base-model properties, not to training-recipe variation.

D Pearson Correlation: Fertility vs. Accuracy Drop

To quantify the claim in Section 5 that tokenizer fertility does not primarily mediate the Latin accuracy drop, we compute Pearson correlations between Δ_{fert} (tokens per word on Latin minus tokens per word on Cyrillic) and Δ_{acc} (accuracy on Cyrillic minus accuracy on Latin) across all (tokenizer, benchmark, Latin variant) measurements. We use nine open-weight tokenizers, three benchmarks, and four Latin variants, yielding $n = 108$ paired observations.

The pooled correlation is $r = +0.36$: weakly positive, in the direction predicted by hypothesis (tokenization explosion), but well short of the value expected if tokenization were the primary driver. The correlation is somewhat lower after excluding Sherkala-8B ($r = +0.26$), indicating that Sherkala contributes to but does not solely produce the as-

Model	Cyr	Latin variant				Lat	Δ	Hint	Map
		2017	2018	2019	2021				
<i>Open-weight models</i>									
Llama 3.2-3B-Instruct	35.3	29.7	29.5	30.1	30.8	30.0	5.3	29.4	29.1
Llama 3.1-8B-Instruct	44.7	30.2	31.3	30.5	32.0	31.0	13.7	25.5	30.8
Gemma-4-E2B-it	44.8	34.4	35.0	35.5	36.6	35.4	9.4	38.5	40.0
Gemma-4-E2B-it (<i>think</i>)	60.3	40.6	44.6	42.8	43.2	42.8	17.5	42.9	43.0
Gemma-4-E4B-it	51.1	35.2	38.8	37.1	38.6	37.4	13.7	43.2	39.6
Gemma-4-E4B-it (<i>think</i>)	69.2	49.3	56.9	54.0	54.4	53.6	15.6	56.1	54.0
Qwen3.5-4B	52.3	36.3	40.9	38.4	40.3	39.0	13.3	38.4	37.6
Qwen3.5-4B (<i>think</i>)	69.4	51.2	59.2	56.2	56.6	55.8	13.6	55.4	50.4
Qwen3.5-9B	60.2	41.2	47.0	44.5	45.7	44.6	15.6	45.0	43.3
Qwen3.5-9B (<i>think</i>)	74.5	58.8	65.9	64.1	64.1	63.2	11.3	63.1	61.2
<i>Kazakh-adapted models</i>									
KazLLM-8B	44.7	31.7	32.3	32.1	33.8	32.5	12.2	31.8	31.7
Sherkala-8B	48.8	31.3	31.9	31.5	34.7	32.4	16.4	31.8	30.8
Qolda	47.1	32.2	34.8	36.0	36.0	34.8	12.4	42.9	42.0
Qolda (<i>think</i>)	50.0	46.3	44.3	45.5	40.3	44.1	5.9	51.2	50.3
<i>Proprietary models</i>									
Gemini-2.5-Flash	78.8	62.3	70.2	67.7	67.9	67.0	11.8	–	–
Gemini-3.1-Flash	87.4	75.1	80.7	78.9	79.2	78.5	9.0	–	–
GPT-4o	76.1	54.5	62.0	62.8	63.7	60.8	15.3	–	–
GPT-5	92.5	92.0	92.2	92.0	92.0	92.1	0.4	–	–

Table 8: Accuracy (%) for the KazMMLU benchmark. *Cyr* refers to the Cyrillic baseline. Columns labeled 2017 to 2021 indicate baseline accuracy for each Latin variant. *Lat* represents the unweighted mean across the four Latin variants. Δ is the difference between *Cyr* and *Lat*. The *Hint* and *Map* columns show the unweighted mean for the four Latin variants under the alphabet-hint and full-mapping prompt conditions. “–” means the model was not evaluated for that scenario. (*think*) indicates a reasoning-mode run.

Model	Cyr	Latin variant				Lat	Δ	Hint	Map
		2017	2018	2019	2021				
<i>Open-weight models</i>									
Llama 3.2-3B-Instruct	30.0	26.0	26.0	28.0	25.8	26.4	3.6	26.4	25.6
Llama 3.1-8B-Instruct	35.0	23.2	25.8	26.1	24.4	24.9	10.1	15.9	25.4
Gemma-4-E2B-it	32.2	25.1	25.6	24.3	25.0	25.0	7.2	24.4	25.1
Gemma-4-E2B-it (<i>think</i>)	36.6	26.3	27.9	26.8	28.9	27.5	9.1	26.4	27.2
Gemma-4-E4B-it	40.6	27.8	30.0	28.0	28.3	28.5	12.1	28.6	27.9
Gemma-4-E4B-it (<i>think</i>)	45.9	29.9	35.2	32.3	33.5	32.7	13.2	32.9	32.0
Qwen3.5-4B	44.2	30.7	32.6	31.5	31.1	31.5	12.7	30.5	30.4
Qwen3.5-4B (<i>think</i>)	48.2	31.0	36.4	35.3	32.2	33.7	14.5	30.7	29.6
Qwen3.5-9B	53.8	37.1	40.6	38.8	37.6	38.5	15.3	36.3	35.1
Qwen3.5-9B (<i>think</i>)	55.0	38.2	43.9	41.5	40.9	41.1	13.9	38.1	37.5
<i>Kazakh-adapted models</i>									
KazLLM-8B	36.4	25.7	26.4	25.6	26.3	26.0	10.4	25.1	25.0
Sherkala-8B	53.8	25.3	26.3	26.9	25.8	26.1	27.7	25.3	25.6
Qolda	56.2	39.4	43.7	43.7	46.5	43.3	12.9	41.5	40.3
Qolda (<i>think</i>)	51.5	36.7	38.9	42.0	42.8	40.1	11.4	39.2	38.0
<i>Proprietary models</i>									
Gemini-2.5-Flash	58.4	39.0	48.8	44.1	43.1	43.8	14.7	–	–
Gemini-3.1-Flash	74.2	55.4	64.9	62.8	62.7	61.5	12.7	–	–
GPT-4o	68.9	42.3	51.0	50.5	52.9	49.2	19.7	–	–
GPT-5	77.3	72.0	74.4	74.6	74.2	73.8	3.5	–	–

Table 9: Accuracy (%) for the KazCulture benchmark. *Cyr* refers to the Cyrillic baseline. Columns labeled 2017 to 2021 indicate baseline accuracy for each Latin variant. *Lat* represents the unweighted mean across the four Latin variants. Δ is the difference between *Cyr* and *Lat*. The *Hint* and *Map* columns show the unweighted mean for the four Latin variants under the alphabet-hint and full-mapping prompt conditions. “–” means the model was not evaluated for that scenario. (*think*) indicates a reasoning-mode run.

Model	Cyr	Latin variant				Lat	Δ	Hint	Map
		2017	2018	2019	2021				
<i>Open-weight models</i>									
Llama 3.2-3B-Instruct	14.0	5.6	7.1	5.4	6.1	6.0	8.0	6.2	5.3
Llama 3.1-8B-Instruct	51.0	9.6	12.4	11.0	12.0	11.2	39.8	13.2	11.7
Gemma-4-E2B-it	68.5	10.3	19.0	13.1	12.9	13.8	54.7	16.1	16.6
Gemma-4-E2B-it (<i>think</i>)	72.8	13.7	24.5	16.8	15.5	17.6	55.2	20.1	21.1
Gemma-4-E4B-it	79.2	28.3	49.6	33.9	34.7	36.6	42.6	48.7	46.5
Gemma-4-E4B-it (<i>think</i>)	82.3	35.0	61.1	48.0	45.8	47.5	34.8	56.2	52.3
Qwen3.5-4B	74.8	32.7	49.2	42.2	42.4	41.6	33.2	56.8	57.3
Qwen3.5-4B (<i>think</i>)	82.6	45.6	69.1	59.3	59.4	58.3	24.2	53.5	59.8
Qwen3.5-9B	88.0	69.2	78.1	76.7	81.1	76.3	11.7	83.4	84.0
Qwen3.5-9B (<i>think</i>)	84.9	51.9	78.2	70.4	70.6	67.8	17.1	69.5	64.9
<i>Kazakh-adapted models</i>									
KazLLM-8B	61.1	11.8	23.5	16.1	21.5	18.2	42.9	16.9	15.8
Sherkala-8B	69.1	17.4	23.4	22.4	25.0	22.0	47.0	23.8	22.0
Qolda	77.9	64.2	68.3	69.6	72.8	68.7	9.2	64.8	65.7
Qolda (<i>think</i>)	82.3	51.2	61.6	61.7	62.5	59.2	23.0	69.2	75.2
<i>Proprietary models</i>									
Gemini-2.5-Flash	89.3	73.5	83.7	80.1	80.6	79.5	9.8	–	–
Gemini-3.1-Flash	92.0	85.7	88.9	88.7	87.7	87.8	4.2	–	–
GPT-4o	90.5	71.4	82.8	84.5	83.1	80.4	10.0	–	–
GPT-5	95.0	94.1	94.3	95.7	95.0	94.8	0.2	–	–

Table 10: Accuracy (%) for the GSM8k-Kazakh benchmark. *Cyr* refers to the Cyrillic baseline. Columns labeled 2017 to 2021 indicate baseline accuracy for each Latin variant. *Lat* represents the unweighted mean across the four Latin variants. Δ is the difference between *Cyr* and *Lat*. The *Hint* and *Map* columns show the unweighted mean for the four Latin variants under the alphabet-hint and full-mapping prompt conditions. “–” means the model was not evaluated for that scenario. (*think*) indicates a reasoning-mode run.

sociation. Per-tokenizer correlations range from +0.15 (Sherkala) to +0.61 (Qolda), with the highest values appearing not for the most Kazakh-adapted tokenizer but for tokenizers (Llama 3.2-3B, Qolda) whose fertility decreases on Latin while accuracy nonetheless drops, confirming that the relationship, where it exists, runs against the tokenization-explosion hypothesis rather than supporting it.

E Statistical Noise Floor

We bound per-cell statistical uncertainty analytically using only benchmark sizes. The bound is conservative and depends on no assumption about the underlying accuracy of any specific model.

Derivation. For a binomial proportion $\hat{p}$ estimated from n independent trials, the Wald standard error is

$$\text{SE}(\hat{p}) = \sqrt{\frac{p(1-p)}{n}}. \quad (2)$$

The 95% Wald confidence interval is $\hat{p} \pm 1.96 \cdot \text{SE}$, with half-width

$$H = 1.96 \cdot \sqrt{\frac{p(1-p)}{n}}. \quad (3)$$

The factor $p(1-p)$ is a downward-opening parabola in p , maximized at $p = 0.5$ where it equals

0.25. For any unknown accuracy $p \in [0, 1]$ the worst-case half-width is therefore

$$H_{\max} = \frac{1.96}{2\sqrt{n}} = \frac{0.98}{\sqrt{n}}, \quad (4)$$

depending only on n .

Per-benchmark bounds. Applying Eq. 4 to the three benchmarks:

- GSM8k-Kazakh ($n = 1,319$): $H_{\max} = \pm 2.70$
- KazCulture ($n = 1,334$): $H_{\max} = \pm 2.68$
- KazMMLU ($n = 9,870$): $H_{\max} = \pm 0.99$

Averaged-across-benchmarks bound. The per-model accuracies reported in Table 1, Table 3, and Table 4 are unweighted means of the three per-benchmark accuracies. Since the three benchmarks evaluate models on disjoint item sets, the per-benchmark estimates are independent binomial samples, and the standard error of their mean is

$$\text{SE}_{\text{avg}} = \frac{1}{3} \sqrt{\text{SE}_1^2 + \text{SE}_2^2 + \text{SE}_3^2}. \quad (5)$$

At the worst case $p = 0.5$ for all three benchmarks, $\text{SE}_{\text{avg}} = 0.669$ percentage points, yielding a worst-case 95% half-width of ± 1.31 points on averaged accuracies.

Setting	Value
<i>OpenAI models</i>	
Models	gpt-4o-2024-08-06, gpt-5
Access date	12 May 2026
Temperature	0.0 (GPT-4o); not sent for GPT-5 (API default)
Max completion tokens	512
Reasoning effort	not set; GPT-5 at its de- fault configuration
<i>Gemini models</i>	
Models	gemini-2.5-flash, gemini-3.1-flash
Access date	11 May 2026
Temperature	not sent; service default applies
Max completion tokens	not sent; service default
Reasoning effort	not set; thinking at the model default

Table 11: API evaluation specifications for proprietary models.

Interpretation. The Cyrillic-to-Latin drops reported in Table 1 range from 5.6 to 30.4 points. All exceed twice the worst-case per-cell bound on the smallest benchmark ($2 \times 2.70 = 5.40$); the smallest margin is Llama-3.2-3B-Instruct ($\Delta = 5.6$), which clears the bound by 0.2 points. Every other model clears it by a wide margin, and on KazMMLU specifically (where $H_{\max} = \pm 0.99$) every reported drop exceeds the worst-case bound by more than an order of magnitude. The cross-script effect is therefore well outside binomial sampling noise.

Smaller pairwise differences require more care. The prompt-level mitigation effects in Table 3 have an average magnitude of approximately 1 point on averaged accuracies, within the averaged-benchmarks noise floor (± 1.31 points). These effects should not be interpreted as reliable improvements; their near-zero average across nine model configurations is consistent with the null hypothesis that hint and mapping interventions have no systematic effect. This reinforces, rather than weakens, the paper’s conclusion that prompt-level patches fail to close the script gap. In contrast, the LoRA effects on the targeted Latin variant in Table 4 (+4 to +20 points) and the Cyrillic-forgetting effects on weaker bases (-3 to -10 points) all exceed the averaged-benchmarks bound and are not noise-driven.

Hyperparameter	Value
<i>LoRA configuration</i>	
LoRA rank (r)	16
LoRA α	32
LoRA dropout	0.05
Target modules	all linear: $q, k, v, o,$ gate, up, down_proj
<i>Optimization</i>	
Optimizer	AdamW
Precision	bf16
Gradient checkpointing	enabled
Learning rate	2×10^{-4} (3B, 4B); 1.5×10^{-4} (8B)
LR schedule	cosine
Warmup ratio	0.03
Epochs	3
Effective batch size	64 (per-device $\times$ grad-accum $\times$ 2 GPUs)
Maximum sequence length	2048
Sequence packing	enabled
Random seed	42
<i>Data and hardware</i>	
Training data	KazCulture train split, 85% Latin / 15% Cyrillic
Training framework	LLaMA-Factory (Zheng et al., 2024)
Hardware (per run)	$2 \times$ NVIDIA H100 80GB, DDP

Table 12: LoRA fine-tuning hyperparameters. Items that vary by base model are listed with the base size in parentheses; all other values are shared across the three base models (Qwen3.5-4B, Llama-3.2-3B, KazLLM-8B). The 85 / 15 Latin / Cyrillic mixing ratio is a fixed design choice rather than a tuned hyperparameter.

Subset	n	Pearson r
All tokenizers	108	+0.36
Excluding Sherkala-8B	96	+0.26
Sherkala-8B only	12	+0.15
<i>Per-tokenizer (12 obs. each)</i>		
Gemma-4-E2B-it	12	+0.19
Gemma-4-E4B-it	12	+0.27
Qwen3.5-4B	12	+0.39
Qwen3.5-9B	12	+0.34
Llama 3.1-8B-Instruct	12	+0.39
Llama 3.2-3B-Instruct	12	+0.57
KazLLM-8B	12	+0.37
Sherkala-8B	12	+0.15
Qolda	12	+0.61

Table 13: Pearson correlation between Δ_{fert} (token fertility change from Cyrillic to Latin) and Δ_{acc} (accuracy drop from Cyrillic to Latin). Per-tokenizer values are computed over 3 benchmarks $\times$ 4 Latin variants = 12 paired observations.