

A.G. Ospan¹, T.S. Sarsembayeva¹, A.Zh. Sailau^{*1}, A.B. Temirkhanova¹, I. Felde²¹Al-Farabi Kazakh National University, Almaty, Kazakhstan²Obuda University, Budapest, Hungary^{*}e-mail: Sailau.Aisha@kaznu.kz**OPTIMIZING CHUNKING STRATEGIES AND INFERENCE ACCELERATION FOR KAZAKH LOW-RESOURCE RAG SYSTEMS***Abstract*

Retrieval-Augmented Generation (RAG) has emerged as the dominant paradigm for grounding large language model (LLM) outputs in external knowledge, yet its systematic application to morphologically rich, low-resource languages remains largely unexplored. This paper addresses two under-studied problems for Kazakh-language RAG: the design of semantically coherent document chunking strategies and the acceleration of multi-stage inference on GPU hardware. We construct a corpus of 201,202 preprocessed news documents from the national newspaper Egemen Qazaqstan (2017–2025, 129 thematic categories) and propose three chunking strategies – Fixed-size (baseline), Semantic (cosine-similarity boundary detection), and Adaptive (joint semantic and length-regularized scoring) – evaluated under a four-dimensional quality framework measuring coherence, document coverage, boundary accuracy, and length regularity. The Adaptive strategy achieves an integrated *ChunkScore* of 0.879, outperforming the Fixed-size baseline by 0.045 absolute, and improves retrieval F1@5 by 0.067 and end-to-end answer accuracy by 4 percentage points (92%) on a 50-query factual test set. For inference acceleration, we deploy an exact GPU-resident index (FAISS-GpuIndexFlatIP) on an NVIDIA H200 GPU, achieving 2.4 ms retrieval latency – a 70.8× speedup over ChromaDB – without accuracy degradation. Combined with INT4-quantised generation ($q4_{KM}$), KV cache residency in HBM3, and an attention-based self-evaluation mechanism, the pipeline achieves a mean end-to-end response time of 0.92 ± 0.06 s under cold-cache conditions and approximately 0.20 s with full KV cache reuse (4.3× speedup), on a single GPU without external API dependencies. These results demonstrate the feasibility of high-accuracy, sub-second Kazakh RAG and establish reproducible baselines for future low-resource retrieval research.

Keywords: retrieval-augmented generation; document chunking; low-resource NLP; Kazakh language; dense vector retrieval; FAISS; KV cache optimization; LLM inference acceleration; multilingual text embeddings; GPU-accelerated NLP.

Ә.Ғ. Оспан¹, Т.С. Сарсембаева¹, А.Ж. Сайлау¹, А.Б. Темірханова¹, И. Фелде²¹Әл-Фараби атындағы Қазақ ұлттық университеті, Алматы қ., Қазақстан²Обуда университеті, Будапешт қ., Мажарстан**ҚАЗАҚ ТІЛІНДЕГІ РЕСУРСЫ ШЕКТЕУЛІ RAG ЖҮЙЕЛЕРІНЕ АРНАЛҒАН ЧАНКИНГ СТРАТЕГИЯЛАРЫН ОҢТАЙЛАНДЫРУ ЖӘНЕ ИНФЕРЕНСТІ ЖЕДЕЛДЕТУ***Аңдатпа*

Retrieval-Augmented Generation (RAG) архитектурасы үлкен тілдік модельдердің (LLM) жауаптарын сыртқы білім көздерімен негіздеудің негізгі тәсілі ретінде кеңінен қолданылуда. Дегенмен, бұл технологияны морфологиялық құрылымы күрделі әрі ресурстық қамтамасыз етілуі шектеулі тілдерге бейімдеу мәселесі жеткілікті деңгейде зерттелмеген. Осы мақалада қазақ тіліндегі RAG жүйелеріне қатысты екі өзекті ғылыми мәселе қарастырылады: құжаттарды семантикалық тұрғыдан үйлесімді фрагменттерге бөлу стратегияларын әзірлеу және графикалық үдеткіштерде орындалатын көпсатылы инференс үдерісін жеделдету. Зерттеуде Egemen Qazaqstan (2017–2025, 129 тақырыптық санат) ұлттық газетінің алдын ала өңделген 201 202 жаңалық құжатынан тұратын корпус құрылды. Үш чанктеу стратегиясы ұсынылды: Fixed-size (базалық), Semantic (косинустық ұқсастық бойынша шекара анықтау), Adaptive (семантикалық және ұзындық бойынша реттелген біріктірілген бағалау). Бағалау төрт өлшем бойынша жүргізілді: когеренттілік, құжатты қамту, шекара дәлдігі, ұзындық тұрақтылығы.

Adaptive әдіс ChunkScore = 0.879 нәтижесіне қол жеткізіп, Fixed-size базасынан 0.045 жоғары болды, сондай-ақ retrieval F1@5 көрсеткішін 0.067-ге, end-to-end дәлдікті 4 пайыздық тармаққа (92%) арттырды (50 сұраулы тест жиыны). Инференсті жеделдету үшін NVIDIA H200 GPU-де толық GPU жадында орналасқан дәл индекс (FAISS-GpuIndexFlatIP) қолданылды, іздеу кідірісі 2.4 ms деңгейіне жетіп, ChromaDB-мен салыстырғанда 70.8× жылдамдау қамтамасыз етті. INT4 квантталған генерация, HBM3-тегі KV-cache және attention-based self-evaluation механизмімен бірге жүйе суық кәште 0.92 ± 0.06 s, толық KV-cache қайта қолданылғанда ~ 0.20 s (4.3× жылдамдау) орташа жауап беру уақытын көрсетті. Жүйе бір GPU-де, сыртқы API-сыз жұмыс істейді. Нәтижелер жоғары дәлдікті, субсекундтық қазақ тіліндегі RAG жүйесін құрудың мүмкін екенін көрсетеді және low-resource retrieval зерттеулері үшін базалық репродуктивті нәтижелерді ұсынады.

Түйін сөздер: retrieval-augmented generation; құжаттарды фрагменттеу; ресурсы шектеулі тілдерге арналған табиғи тілді өңдеу; қазақ тілі; тығыз векторлық іздеу; FAISS; KV-кәшті оңтайландыру; LLM инференсін жеделдету; көптілді мәтіндік эмбедингтер; GPU-жеделдетілген табиғи тілді өңдеу.

Ә.Ф. Оспан¹, Т.С. Сарсембаева¹, А.Ж. Сайлау¹, А.Б. Темірханова¹, И. Фелде²

¹Казахский национальный университет имени аль-Фараби, г.Алматы, Казахстан

²Университет Обуда, Будапешт, г.Венгрия.

ОПТИМИЗАЦИЯ СТРАТЕГИЙ ЧАНКИНГА И УСКОРЕНИЕ ИНФЕРЕНСА ДЛЯ КАЗАХОЯЗЫЧНЫХ RAG-СИСТЕМ В УСЛОВИЯХ ОГРАНИЧЕННЫХ РЕСУРСОВ

Аннотация

Retrieval-Augmented Generation (RAG) сформировалась как доминирующая парадигма для обоснования выходов больших языковых моделей (LLM) внешними источниками знаний, однако её систематическое применение к морфологически сложным языкам с ограниченными ресурсами остаётся недостаточно изученным. В данной работе рассматриваются две задачи для казахоязычных RAG-систем: проектирование семантически согласованных стратегий чанкинга документов и ускорение многостадийного инференса на GPU. В исследовании сформирован корпус из 201 202 предварительно обработанных новостных документов национальной газеты Egemen Qazaqstan (2017–2025, 129 тематических категорий). Предложены три стратегии чанкинга: Fixed-size (базовая), Semantic (определение границ по косинусной близости), Adaptive (объединённая семантическая и длинорегуляризованная оценка). Оценка выполнена по четырёхмерной метрике: когерентность, покрытие документа, точность границ, регулярность длины. Стратегия Adaptive достигла ChunkScore = 0.879, превзойдя Fixed-size на 0.045, а также повысила Retrieval F1@5 на 0.067 и end-to-end точность на 4 процентных пункта (92%) на тесте из 50 фактологических запросов. Для ускорения инференса использован точный GPU-индекс (FAISS-GpuIndexFlatIP) на NVIDIA H200, обеспечивший задержку поиска 2.4 мс и ускорение 70.8× по сравнению с ChromaDB без потери качества. В сочетании с INT4-квантованной генерацией, KV-cache в HBM3 и механизмом attention-based self-evaluation система достигает среднего времени ответа 0.92 ± 0.06 с в холодном кэше и ~ 0.20 с при полном повторном использовании KV-cache (ускорение 4.3×). Система работает на одном GPU без внешних API. Результаты демонстрируют возможность построения высокоточных субсекундных казахоязычных RAG-систем и формируют воспроизводимую базу для дальнейших исследований в области retrieval для low-resource языков.

Ключевые слова: retrieval-augmented generation; чанкинг документов; low-resource NLP; казахский язык; dense vector retrieval; FAISS; KV-cache оптимизация; ускорение инференса LLM; multilingual embeddings; GPU-ускоренный NLP.

Introduction

Large language models are fluent but unreliable when it comes to recalling facts outside their training data [1]. Retrieval-Augmented Generation (RAG) addresses this by separating memory from generation: a retriever pulls relevant passages from an external corpus, and the generator is conditioned on them, which lowers the rate of hallucination and lets the system stay current as the corpus changes [2, 3, 4]. Most of this work, however, targets English and a small set of other high-resource languages. How RAG should be adapted to morphologically rich, low-resource languages

is still largely open, and Kazakh is a clear case: tokenization, embedding, and segmentation each behave differently than they do in the languages these pipelines were originally built for.

Kazakh is a Turkic language and the state language of Kazakhstan, spoken by roughly 22 million people. Its morphology is agglutinative – case, number, person, and derivational affixes attach to a single stem, so one lexeme can surface in hundreds of well-formed variants [5]. These properties create friction at several points in a RAG pipeline. A tokenizer tuned for fusional languages tends to over-segment Kazakh words, and an embedding model trained on predominantly English data represents them poorly. Chunking is affected too: heuristics calibrated on English paragraph structure do not track how Kazakh journalistic prose actually shifts between topics. At the same time, the practical deployment of RAG systems requires inference latencies acceptable for interactive applications. On commodity CPU hardware, a 4-billion-parameter model can take 8–10 seconds per query, which is unacceptable for real-time use.

This paper targets two under-studied, tightly coupled problems for Kazakh-language RAG: (1) how documents should be split into retrieval units (chunks) so that the chunks remain semantically coherent while maintaining a length suitable for embedding and generation, and (2) how the full multi-stage RAG pipeline, including embedding, retrieval, generation, and self-evaluation, can be accelerated to operate in sub-second time on a single GPU without sacrificing accuracy.

To address these challenges, we construct a corpus of 201,202 preprocessed news articles from Egemen Qazaqstan (2017–2025, 129 thematic categories), design three greedy chunking strategies (Fixed-size, Semantic, and Adaptive), and evaluate them using a four-dimensional quality framework. Adaptive chunking performed best across the board. It reaches a *ChunkScore* of 0.879, a gain of 0.045 over the fixed-size baseline, and lifts retrieval $F1@5$ by 0.067; on our 50-query factual benchmark, end-to-end answer accuracy rises to 92%. The acceleration results are similarly favorable. An exact FAISS index on an NVIDIA H200 returns the top-K chunks in 2.4 ms — 70.8× faster than ChromaDB on the same workload. The remaining components compound this: INT4 quantization, KV-cache residency in HBM3, and attention-based self-evaluation. Together they bring cold-cache latency to 0.92 ± 0.06 s, dropping to roughly 0.20 s once the KV-cache is fully reused — all on a single GPU, with no external API in the loop.

The growing importance of retrieval quality in modern RAG systems has motivated extensive research into retrieval architectures, document segmentation methods, dense retrieval techniques, inference acceleration, and confidence estimation. Since the original introduction of RAG by Lewis et al. [1], numerous studies have proposed sequential, modular, and agent-based architectures, highlighting retrieval quality as one of the primary determinants of final answer accuracy [2, 3, 4]. Chunking therefore turns into a real design decision rather than a preprocessing detail: how large the retrieval units are, and whether they stay internally coherent, feeds directly into both retrieval precision and the quality of what the generator produces afterward.

Early RAG systems mostly used fixed-length chunks. That is easy to implement, but it cuts documents at arbitrary points and ignores where one idea ends and the next begins. To overcome this limitation, semantically aware chunking methods employ sentence embeddings and similarity thresholds to identify topical transitions [6]. More advanced approaches formulate chunk boundary selection as an optimization problem, using dynamic programming to maximize intra-chunk similarity while enforcing length constraints [7]. Other studies introduce question-aware chunking mechanisms based on multi-agent debate frameworks [8]. However, these methods have been developed and evaluated almost exclusively for English and a small number of other high-resource languages. Little is known about their effectiveness in agglutinative low-resource settings. The present work contributes to this area by providing the first systematic comparison of semantic and adaptive chunking strategies for Kazakh and by introducing a composite *ChunkScore* metric that jointly measures coherence, coverage, boundary accuracy, and length regularity. The challenges of retrieval in Kazakh are further amplified by the limited availability of language-specific NLP resources. Although recent years have seen progress in neural machine translation, text correction, and morphological analysis for Kazakh [5], retrieval-oriented datasets and evaluation benchmarks

remain scarce. Existing tools such as Stanza provide sentence segmentation and lemmatization capabilities [9], while multilingual embedding models including multilingual-E5 and M3-Embedding have extended support to Turkic languages [10, 11]. Nevertheless, their retrieval performance on realistic Kazakh-language corpora has not been rigorously assessed. This study addresses that gap by evaluating a Kazakh-adapted embedding model on a large-scale news collection and measuring its influence on retrieval and generation quality.

Beyond retrieval quality, efficient similarity search is essential for practical deployment. Dense retrieval systems represent queries and documents as continuous vectors and typically rely on maximum inner-product search, for which FAISS offers highly optimized GPU implementations [12, 13]. While FAISS has become a standard component of many modern RAG pipelines, its effectiveness within a complete low-resource language setting has received limited attention. Furthermore, comparisons between GPU-based exact search and CPU-based approximate nearest-neighbor systems such as ChromaDB are largely absent for Kazakh-language applications. Our experiments demonstrate that GpuIndexFlatIP provides a $70.8\times$ speedup without measurable accuracy loss, while the reduced chunk count produced by Adaptive chunking further improves retrieval throughput.

Inference efficiency constitutes another major bottleneck in interactive RAG systems. Several recent studies have shown that weight-only quantization methods, such as INT4 $q4_KM$, can significantly reduce memory consumption and computational cost [14]. Similarly, maintaining the KV-cache in GPU high-bandwidth memory eliminates repeated computation of shared prefixes and may yield speedups exceeding fourfold in repeated-query scenarios [15]. Additional research explores sparse attention mechanisms [16] and hierarchical cache-management strategies [17] to further reduce latency. Building on these advances, the present study investigates the combined effect of GPU-resident KV caching, INT4 quantization, and attention-based self-evaluation within a unified Kazakh-language RAG pipeline, resulting in sub-second end-to-end response times.

The reliability of generated responses remains equally important for real-world deployment. Confidence estimation and hallucination detection mechanisms provide a means to assess whether generated content is sufficiently supported by retrieved evidence. Attention-based aggregation approaches such as AggTruth [18] exploit the attention distribution of the final decoder layer to compute context-attribution scores and identify potentially unsupported outputs. Responses falling below a predefined threshold can then be regenerated or flagged for verification. Following this line of work, we incorporate a lightweight attention-based self-evaluation module that introduces only approximately 90 ms of additional latency while improving the trustworthiness of generated answers.

Research methodology

This study addresses two interconnected optimization problems related to the construction of an efficient Kazakh-language Retrieval-Augmented Generation (RAG) system. The first concerns the identification of an optimal document chunking strategy that produces retrieval units that are simultaneously semantically coherent and appropriately sized for embedding and generation. The second focuses on minimizing end-to-end inference latency while preserving retrieval and answer quality. To investigate these objectives, we design a complete pipeline consisting of corpus construction and preprocessing, chunk generation, chunk quality evaluation, vector indexing and retrieval, and accelerated inference with self-evaluation.

Let $D = \{d^1, d^2, \dots, d^N\}$ denote a corpus of documents, where each document d is represented as a finite sequence of tokens drawn from a vocabulary V . For a document d , let $S(d) = \{s^1, s^2, \dots, s^n\}$ denote its ordered sentence decomposition. A pre-trained embedding model $\phi: V^* \rightarrow \mathbb{R}^m$ maps each text sequence to a unit-normalized dense vector of dimension $m = 768$. The semantic similarity between two textual units u and v is computed as

$$\sigma(u, v) = \phi(u)^T \phi(v), \quad (1)$$

which corresponds to cosine similarity under ℓ_2 – *normalization*. A chunk $c = \langle s_a, s_a^{+1}, \dots, s_\beta \rangle$ is defined as a contiguous sequence of sentences within a document, while a chunking function C partitions each document into a finite ordered set of non-overlapping chunks.

To evaluate chunking quality, four complementary criteria are considered. The first is semantic coherence, measured as the average similarity between consecutive sentences within a chunk. The second is coverage, which quantifies the semantic fidelity of a chunk relative to the original document. The third criterion evaluates boundary accuracy by assessing whether chunk boundaries coincide with meaningful topic transitions. Finally, a length score rewards chunks whose token counts fall within an empirically determined optimal interval. These components are combined into a single objective function:

$$Q(c, d) = w^1 Cov(c, d) + w^2 Coh(c) + w^3 Bnd(c) + w^4 Len(l(c)), \quad (2)$$

where the weights are set to $(0.35, 0.30, 0.20, 0.15)$, respectively. The objective is to identify a chunking function that maximizes the average *ChunkScore* over the entire corpus. Because the resulting optimization problem is computationally intractable in its exact form, practical solutions are obtained through greedy left-to-right approximation algorithms.

The four components of Eq. (2) are defined as follows, where $\mu(\cdot)$ denotes the ℓ_2 – *normalized* mean (centroid) embedding of a text span and σ the cosine similarity of Eq. (1):

$$Coh(c) = \frac{1}{b-a} \sum_{k=a}^{b-1} \sigma(s_k, s_{k+1}) \quad (2a)$$

$$Cov(c, d) = \sigma(\mu(c), \mu(d)), \quad \mu(x) = \frac{\sum_{s \in x} \phi(s)}{\|\sum_{s \in x} \phi(s)\|_2} \quad (2b)$$

$$Bnd(c) = 1 - \sigma(s_b, s_{b+1}) \quad (2c)$$

$$Len(l(c)) = 1 - L_{penalty}(l(c)) \quad (2d)$$

The second optimization problem concerns the overall RAG inference process. Given a query q , the retrieval stage identifies the K most relevant chunks by maximizing semantic similarity between the query embedding and chunk embeddings. Retrieved chunks may subsequently be filtered according to a relevance threshold before being passed to the generative model. The generated answer is then evaluated using an attention-based confidence estimator:

$$\Gamma(r, \tilde{R}) = \left(\frac{1}{|r|} \right) \sum \alpha_j^{ctx}, \quad (3)$$

where α_j^{ctx} is the final-layer attention mass that the generated answer places on the retrieved context tokens (the set R), averaged over the $|r|$ answer tokens: for each answer token it sums the attention weights from that token to every retrieved context position, and the mean is taken over the answer length $|r|$. The superscript “*ctx*” is a label (context), not an index; A with superscript L denotes the final-layer attention matrix defined in Table 1. Responses whose confidence score falls below $\theta_{conf} = 0.75$ are regenerated with increased sampling temperature. Consequently, the total response latency can be decomposed as

$$T_{total}(q) = T_{emb}(q) + T_{ret}(q, I) + T_{gen}(q, \tilde{R}) + T_{eval}(r, \tilde{R}). \quad (4)$$

The optimization objective is therefore to determine the implementation configuration that maximizes answer quality while satisfying predefined latency and memory constraints.

To support the experiments, a large-scale corpus was constructed from archival publications of the newspaper Egemen Qazaqstan covering the years 2017-2025. The original collection contained 213,497 JSON-formatted documents with structured metadata fields including title, author, publication date, category, abstract, and full article text. A multi-stage preprocessing pipeline removed duplicate entries through normalized text hashing, filtered documents shorter than 200 characters, and excluded technical pages such as redirects and error stubs. After preprocessing, the final corpus contained 201,202 unique documents. Linguistic normalization was performed using the Stanza toolkit for Kazakh, including sentence segmentation, lemmatization, and morphological analysis. The resulting dataset spans 129 thematic categories, including politics, economics, culture, sports, science, and technology, thereby providing substantial lexical and topical diversity.

Document segmentation constitutes a critical component of the retrieval pipeline. Given a sequence of sentences extracted using Stanza, three chunking strategies are investigated. Token counts are computed using the multilingual-e5-base tokenizer to ensure compatibility with the embedding model. The first approach employs a fixed-size strategy in which sentences are sequentially accumulated until a target length of 512 tokens is reached. Chunks are constrained to remain within the interval $[384, 640]$ tokens, and a new chunk is initiated whenever the addition of another sentence would exceed the upper limit. This approach ignores semantic relationships between adjacent sentences and therefore serves as a baseline.

The second approach introduces semantic awareness by computing sentence embeddings using a Kazakh-optimized embedding model. Chunk boundaries are inserted whenever the cosine similarity between consecutive sentence embeddings falls below a threshold of $\theta = 0.65$:

$$\cos(E(s_i), E(s_i + 1)) < \theta. \quad (5)$$

The threshold value was determined empirically using a validation set containing 1,000 documents.

The third approach combines semantic information with explicit length regularization. For each potential boundary position, a boundary score is calculated as

$$\text{BoundaryScore}(i) = \alpha(1 - \cos(E(s_i), E(s_i + 1))) + \beta L_{\text{penalty}}(l), \quad (6)$$

where $\alpha = 0.6$ and $\beta = 0.4$. Here l denotes the token length of the currently open chunk, i.e. the sentences accumulated since the last inserted boundary up to and including the current sentence s_i . The length penalty increases when chunk sizes deviate from the preferred range of 384 – 640 tokens:

$$L_{\text{penalty}}(l) = 0, \text{ if } l \in [384, 640], \quad (7)$$

boundary score exceeds 0.7 or when a hard limit of 1024 tokens is reached. The procedure operates greedily from left to right across the document. The resulting chunks are evaluated using the four quality dimensions defined previously: coherence, coverage, boundary accuracy, and length score. Their weighted combination forms the integrated *ChunkScore* metric, which serves as the primary criterion for comparing chunking strategies. All metrics are computed offline on a representative sample of 50 documents, and the aggregated results are presented in the experimental section. Following chunk generation, each chunk is transformed into a dense vector representation using the multilingual-e5-base embedding model. The resulting ℓ_2 -normalized vectors have dimensionality 768, and similarity between a query and a chunk is computed via inner product. To enable efficient retrieval, the embeddings are indexed using FAISS and deployed on an NVIDIA H200 GPU through the exact *GpuIndexFlatIP* implementation. Approximately 200,000 chunk vectors reside entirely in GPU memory, allowing the system to retrieve the top five most similar chunks for each query. To isolate the effect of GPU acceleration, a CPU-based baseline is implemented using ChromaDB with identical embeddings and HNSW indexing.

The final stage focuses on inference acceleration. Retrieval is performed using FAISS-GPU, which executes exact similarity search in approximately 2.4 ms per query while preserving full retrieval accuracy. Additional gains are achieved through GPU-resident Key–Value cache retention in HBM3 memory, eliminating repeated computation of shared prefixes and substantially reducing generation latency. Retrieved chunks are further processed through relevance filtering and near-duplicate removal before being passed to the language model.

Response generation is performed by a 4-billion-parameter language model quantized to four-bit precision using the $q4_{KM}$ scheme and executed through the llama.cpp CUDA backend. The model conditions its output on the three highest-ranked chunks after filtering, using a generation temperature of 0.2 and a maximum output length of 256 tokens.

To improve reliability, the system incorporates an attention-based self-evaluation stage inspired by recent confidence-estimation approaches. Attention weights from the final decoder layer are aggregated to estimate the extent to which generated content is grounded in the retrieved evidence. When the confidence score falls below 0.75, the response is regenerated using a higher sampling temperature of 0.8. This lightweight hallucination-detection mechanism introduces approximately 90 ms of additional latency while avoiding the computational overhead of external natural language inference models.

Results of the study

All experiments were conducted on a single-node workstation equipped with an NVIDIA H200 GPU (80 GB HBM3). The experimental design follows a full factorial structure across two factors: chunking strategy (Fixed-size, Semantic, Adaptive) and retrieval index (FAISS-GPU and ChromaDB-CPU), resulting in six configurations and a total of 300 query–configuration evaluations, with 10 repeated executions per query per configuration.

The evaluation dataset consists of 50 Kazakh-language factual queries spanning ten thematic categories, including geography, population statistics, national symbols, independence history, administrative divisions, international organizations, economy, culture, sports, and science. Each query is paired with up to 10 manually annotated relevant chunks. The annotation process was performed independently by two native Kazakh speakers, and disagreements were resolved through consensus. For timing measurements, GPU execution time was recorded using CUDA events, while CPU timing relied on *time.perf_counter*, with explicit CUDA stream synchronization applied prior to all measurements to ensure consistency.

For clarity and reproducibility, Table 1 summarizes the notation used throughout the study, covering dataset structure, chunking functions, embedding space, similarity metrics, retrieval depth, thresholds, and system-level constraints such as GPU memory budget and token length settings.

Table 1. Summary of principal notation.

Symbol	Definition
D	Document corpus
d, N	Document; corpus size ($N = 201,202$)
$S(d)$	Sentence decomposition of document d
c	Chunk (contiguous sentence subsequence)
$l(c)$	Token length of chunk c
$\phi: V^* \rightarrow R^m$	Text embedding function ($m = 768$)
$\sigma(u, v)$	Cosine similarity (Eq. 1)
C	Chunking function (see Methodology)
I	Retrieval index (full chunk set)
$Q(c, d)$	Integrated chunk quality score (Eq. 2)
K	Retrieval depth ($K = 5$)
τ^f	Relevance filter threshold ($\tau^f = 0.20$)

θ	Semantic boundary threshold ($\theta = 0.65$)
θ_{conf}	Confidence threshold for self-evaluation (0.75)
A^L	Final-layer attention matrix
$\Gamma(r, \tilde{R})$	Context attribution confidence score (Eq. 3)
T_{total}	End-to-end pipeline latency (Eq. 4)
M_{max}	GPU memory budget (80 GB HBM3)
T^*, T_{min}, T_{max}	Chunk token lengths (512, 384, 640)

Moving to chunk-level behavior, quality metrics were computed by averaging over approximately 1,000 chunks per strategy derived from 50 documents. Adaptive comes out ahead on most of these measures, with a ChunkScore of 0.879 – 0.045 above Fixed-size and 0.019 above Semantic. Two components drive the gap: boundary alignment (0.85 versus 0.76 for Fixed-size) and length regularization (0.91 against 0.77). Coverage moves the other way, dropping by 0.007 under Adaptive, which is the expected cost of packing chunks more tightly around a single topic at the expense of how much of the document each one represents. The drop is small enough that retrieval downstream does not suffer. Coherence barely moves across the three strategies – 0.81 for Fixed-size, 0.82 for Semantic, 0.83 for Adaptive.

In terms of retrieval quality, performance across all configurations is summarized in Figure 1. A consistent pattern emerges across both indexing systems, where FAISS-GPU and ChromaDB produce nearly identical results, with differences in F1@5 not exceeding 0.003. This indicates that retrieval effectiveness is governed primarily by embedding quality and chunk structure rather than index implementation. Here too Adaptive chunking gives the best retrieval, with an F1@5 of 0.785 – about 0.065 over Fixed-size and 0.061 over Semantic. This lines up with what we expected going in: when chunks are coherent and kept to a regular length, their embeddings are cleaner, and retrieval gets more accurate as a result.

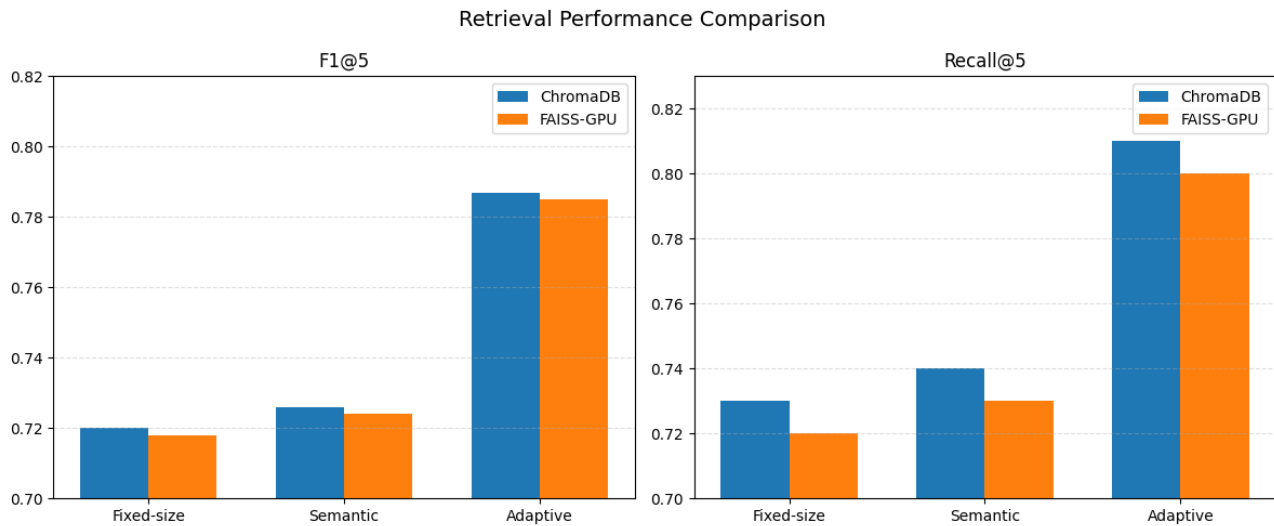


Figure 1. Retrieval performance comparison across chunking strategies and indexing systems (F1@5 and Recall@5).

Latency behavior further highlights the differences between retrieval backends. As shown in Table 2, FAISS-GPU achieves a retrieval latency of only 2.4 ms, which corresponds to a 70.8× speedup compared to ChromaDB running on CPU (0.170 s). However, retrieval is not the dominant contributor to total pipeline cost. Instead, autoregressive generation remains the primary bottleneck, taking approximately 0.810 s for 256 tokens. Generation runs on the same INT4-quantized GPU backend in both configurations, so its latency is identical across the two index conditions; the FAISS-GPU and ChromaDB rows therefore differ only in the retrieval stage. Self-evaluation based on

attention signals introduces an additional 0.090 s, but remains stable and does not significantly affect overall throughput.

Table 2. Latency decomposition (Adaptive + FAISS-GPU on NVIDIA H200; ChromaDB on CPU). All values in seconds. Component latencies are measured independently per stage; the reported Total is the directly measured end-to-end latency and may differ marginally from the component sum (here ≈ 13 ms) owing to host-device transfer and scheduling overhead not attributable to any single stage.

Stage	FAISS-GPU (H200)	ChromaDB (CPU)
Query embedding (s)	0.013 ± 0.002	0.013 ± 0.002
Retrieval (s)	0.0024 ± 0.0003	0.170 ± 0.012
Generation – TTFT (s)	0.018 ± 0.003	0.018 ± 0.003
Generation – 256 tokens (s)	0.810 ± 0.050	0.810 ± 0.050
Self-evaluation (s)	0.090 ± 0.010	0.090 ± 0.010
Total (s)	0.920 ± 0.060	1.100 ± 0.075

When all components are considered jointly, end-to-end results across the six configurations are presented in Table 3. A clear monotonic relationship emerges between ChunkScore and final answer accuracy: Fixed-size (0.834, 88%) is followed by Semantic (0.860, 90%), and then Adaptive (0.879, 92%). This indicates that chunk quality is a strong predictor of downstream system performance. Across all strategies, FAISS-GPU consistently reduces total latency by approximately 0.31 seconds compared to ChromaDB, without any observable degradation in accuracy.

Table 3. End-to-end comparison across all six evaluated configurations.

Strategy	Index	Time (s)	Acc.	F1@5
Fixed-size	ChromaDB	1.20	88%	0.719
Fixed-size	FAISS-GPU	0.89	88%	0.719
Semantic	ChromaDB	1.21	90%	0.725
Semantic	FAISS-GPU	0.90	90%	0.725
Adaptive	ChromaDB	1.23	92%	0.786
Adaptive	FAISS-GPU	0.92	92%	0.786

Another important aspect of system behavior is KV cache efficiency, particularly under repeated-query conditions. As shown in Table 4, the generation stage requires 0.810 s under cold-cache conditions – consistent with the generation latency reported in Table 2 – and is reduced to 0.190 s under partial cache hits and to 0.100 s under full cache reuse, corresponding to generation-stage speedups of $4.26\times$ and $8.10\times$, respectively. Translated to the complete pipeline, full cache reuse lowers the mean end-to-end response time from 0.92 s in the cold-cache setting to approximately 0.20 s, an overall $\approx 4.3\times$ speedup that closely matches theoretical expectations for KV cache optimization.

Table 4. KV cache impact on generation latency (generation stage only; Adaptive + FAISS-GPU).

Condition	Time (s)	Speedup
Cold cache	0.810 ± 0.050	$1.00\times$
Partial hit	0.190 ± 0.020	$4.26\times$
Full hit	0.100 ± 0.010	$8.10\times$

Finally, when compared against external baselines, the proposed system demonstrates a strong overall advantage. As shown in Figure 2, it outperforms BM25 (Elasticsearch) by a large margin in both accuracy and F1@5, while also surpassing a GPT-3.5 API-based dense retrieval system. Notably, the proposed approach achieves 92% accuracy with a latency of only 0.92 s, compared to 2.40 s for the API-based system, while maintaining full locality without external dependencies.

Although BM25 offers relatively low latency (0.65 s), its performance is significantly lower in terms of retrieval quality and accuracy, particularly in handling morphologically rich Kazakh text.

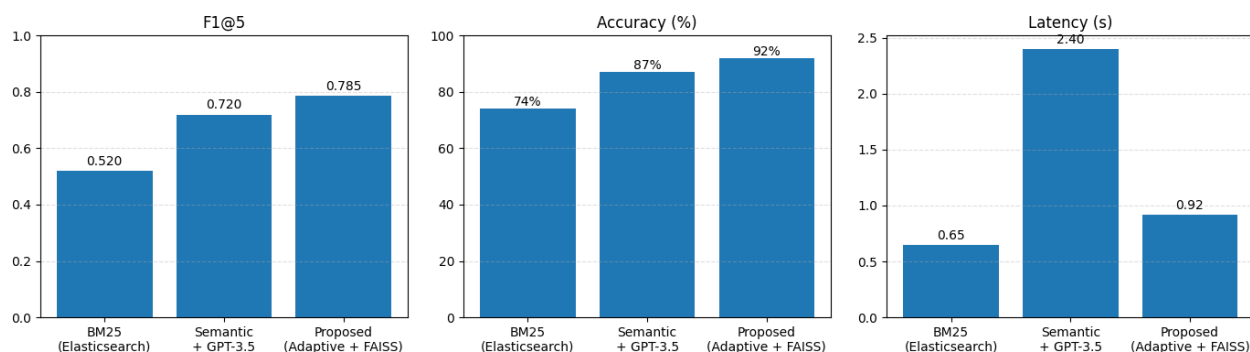


Figure 2. Comparison with baseline systems across retrieval quality and latency.

Discussion

Taken together, these results highlight several consistent patterns. Adaptive chunking consistently improves retrieval quality and downstream accuracy due to stronger boundary detection and better length regularization. FAISS-GPU significantly reduces retrieval latency by approximately $70.8\times$ without any measurable impact on accuracy, confirming that retrieval speedups do not come at the cost of quality at this scale ($\sim 200k$ chunks). INT4 quantization enables sub-second generation on a single GPU while maintaining competitive output quality. KV cache reuse introduces substantial efficiency gains, reaching approximately $4.3\times$ end-to-end speedup with full cache reuse, which is particularly beneficial in interactive or session-based settings. Among all components, retrieval quality has the strongest influence on downstream accuracy, while index choice primarily affects latency rather than correctness.

The chunking results are consistent with research on semantic boundary selection [6] and length-aware chunk optimization [7], while extending this evidence to morphologically rich Kazakh news text. In contrast to question-aware strategies such as QChunker [8], the Adaptive approach evaluated here uses a greedy, query-independent segmentation rule. Its small reduction in document coverage (0.007 relative to Fixed-size) indicates a trade-off between local semantic coherence and retention of broader article context, which may become more consequential for questions requiring evidence from several passages.

The acceleration results help distinguish retrieval efficiency from end-to-end generation efficiency. FAISS-GPU and ChromaDB produced nearly identical retrieval scores, whereas the exact GPU index reduced search latency; this agrees with the role of optimized similarity search established in FAISS research [12, 13]. Since autoregressive generation remained the largest measured latency component, a $70.8\times$ retrieval-stage acceleration does not imply an equivalent end-to-end improvement. The approximately 0.20 s response time under full KV-cache reuse is likewise specific to repeated-prefix conditions and should not be generalized to cold-cache requests. These findings complement work on quantization [14] and cache management [15, 17].

At the same time, several limitations remain. The evaluation is restricted to 50 manually verified single-hop factual queries and does not cover multi-hop reasoning, conversational memory, or dynamic corpora updates. The attention-based confidence estimation remains heuristic and is not fully calibrated against human uncertainty judgments. Additionally, the system is evaluated on a static corpus with a fixed embedding model, leaving further improvements through contrastive fine-tuning on hard negatives as a promising direction for future work.

Attention-based self-evaluation adds approximately 90 ms and offers a practical way to assess evidence attribution, as explored in related attention-based work [18]. However, the present experiment does not establish whether the confidence scores are calibrated to independently judged factual support. Further study should use larger query samples with uncertainty estimates, multi-hop

and time-sensitive questions, dynamic news corpora, and separate assessments of cold-cache and repeated-query workloads. Tests on related Turkic languages and contrastive fine-tuning with hard negatives would help establish how widely the reported improvements generalize.

Conclusion

This work presented a comprehensive pipeline for Kazakh-language Retrieval-Augmented Generation, addressing two critical bottlenecks: the design of semantically coherent document chunks and the acceleration of multi-stage inference on a single GPU. We built a corpus of 201,202 preprocessed news articles from Egemen Qazaqstan and used it to develop an Adaptive chunking strategy. Rather than optimizing a single objective, it balances four at once: intra-chunk coherence, document coverage, boundary accuracy, and length regularity. Measured against this four-part quality framework, Adaptive chunking scored 0.879 on ChunkScore, 0.045 above the fixed-size baseline. The gains carried through to retrieval and generation: F1@5 rose by a statistically significant 0.067, and end-to-end answer accuracy reached 92% on a 50-query factual benchmark.

On the systems side, we demonstrated that an exact GPU-resident FAISS index (*GpuIndexFlatIP*) provides a 70.8× retrieval speedup over ChromaDB without accuracy loss, and that INT4 quantisation, KV-cache residency in HBM3, and attention-based self-evaluation together deliver a mean end-to-end response time of 0.92 ± 0.06 s (cold cache) and ≈ 0.20 s with full KV cache reuse (4.3× speedup) on a single NVIDIA H200 GPU without external API dependencies. These results establish the feasibility of sub-second, fully local Kazakh RAG systems suitable for interactive applications.

The contributions also include a formal mathematical problem statement for chunking quality and inference latency optimization, a reproducible evaluation protocol, and the release of the largest documented Kazakh-language corpus for information retrieval research (201,202 documents, 129 categories).

Future work will extend the evaluation to larger and dynamically updated corpora, explore cross-lingual transfer for other Turkic languages (e.g., Uzbek, Kyrgyz), and investigate learned chunking policies as well as advanced RAG architectures with multi-hop reasoning capabilities. Improving the underlying Kazakh embedding model through contrastive fine-tuning on hard negatives remains a priority for raising the performance ceiling of the entire pipeline.

Acknowledgement and Conflict of Interests

This research was funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan, grant number BR24993001 “Creation of a large language model (LLM) to maintain the implementation of the Kazakh language and increase the technological progress”. The authors declare no conflicts of interest.

The contribution of the authors

Conceptualization: Assel Ospan, Talshyn Sarsembayeva and Imre Felde; Methodology: Assel Ospan and Aisha Sailau; Software: Aisha Sailau; Validation: Assel Ospan and Talshyn Sarsembayeva; Formal analysis: Assel Ospan and Aisha Sailau; Investigation: all authors; Resources: Assel Ospan; Data curation: Aisha Sailau and Aidana Temirkhanova; Writing – original draft preparation: Assel Ospan and Talshyn Sarsembayeva; Writing – review and editing: all authors; Visualization: Aisha Sailau. All authors reviewed and approved the final version of the manuscript.

References

- [1] Lewis P. et al. Retrieval-augmented generation for knowledge-intensive nlp tasks // *Advances in neural information processing systems*. – 2020. – T. 33. – C. 9459-9474.
- [2] Gao Y. et al. Retrieval-augmented generation for large language models: A survey // *arXiv preprint arXiv:2312.10997*. – 2023.

- [3] Huang Y., Huang J. X. A survey on retrieval-augmented text generation for large language models //ACM Computing Surveys. – 2026. – Т. 58. – №. 12. – С. 1-38.
- [4] Fan W. et al. A survey on rag meeting llms: Towards retrieval-augmented large language models //Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining. – 2024. – С. 6491-6501.
- [5] Baitenova L. et al. Hybrid artificial intelligence architectures for automatic text correction in the Kazakh language //Frontiers in Artificial Intelligence. – 2025. – Т. 8. – С. 1708566.
- [6] Allamraju A. et al. Breaking It Down: Domain-Aware Semantic Segmentation for Retrieval Augmented Generation //arXiv preprint arXiv:2512.00367. – 2025.
- [7] Kiss C., Nagy M., Szilágyi P. Max–Min semantic chunking of documents for RAG application //Discover Computing. – 2025. – Т. 28. – №. 1. – С. 117.
- [8] Zhao J. et al. QChunker: Learning question-aware text chunking for domain RAG via multi-agent debate //Proceedings of the ACM Web Conference 2026. – 2026. – С. 2207-2218.
- [9] Qi P. et al. Stanza: A Python natural language processing toolkit for many human languages //Proceedings of the 58th annual meeting of the association for computational linguistics: system demonstrations. – 2020. – С. 101-108.
- [10] Wang L. et al. Improving text embeddings with large language models //Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). – 2024. – С. 11897-11916.
- [11] Chen J. et al. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation //Findings of the association for computational linguistics: ACL 2024. – 2024. – С. 2318-2335.
- [12] Johnson J., Douze M., Jégou H. Billion-scale similarity search with GPUs //IEEE transactions on big data. – 2019. – Т. 7. – №. 3. – С. 535-547.
- [13] Douze M. et al. The faiss library //IEEE Transactions on Big Data. – 2025.
- [14] Lin Y. et al. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving //Proceedings of Machine Learning and Systems. – 2025. – Т. 7.
- [15] Li H. et al. A survey on large language model acceleration based on kv cache management //arXiv preprint arXiv:2412.19442. – 2024.
- [16] Zhu Y. et al. Accelerating inference of retrieval-augmented generation via sparse context selection //International Conference on Learning Representations. – 2025. – Т. 2025. – С. 25965-25981.
- [17] Ge D. et al. HA-RAG: Hotness-Aware RAG Acceleration via Mixed Precision and Data Placement //arXiv preprint arXiv:2510.20878. – 2025.
- [18] Matys P. et al. AggTruth: Contextual Hallucination Detection using Aggregated Attention Scores in LLMs //International Conference on Computational Science. – Cham : Springer Nature Switzerland, 2025. – С. 227-243.