An End-to-End Ukrainian RAG for Local Deployment. Optimized Hybrid Search and Lightweight Generation

Mykola Trokhymovych
Pompeu Fabra University
mykola.trokhymovych@upf.edu

Yana Oliinyk
Independent Researcher
oliinykyana@gmail.com

Nazarii Nyzhnyk
Independent Researcher
nazar.nyzhnyk@gmail.com

Abstract

This paper presents a highly efficient Retrieval-Augmented Generation (RAG) system built specifically for Ukrainian document question answering, which achieved 2nd place in the UNLP 2026 Shared Task.[1] Our solution features a custom two-stage search pipeline that retrieves relevant document pages, paired with a specialized Ukrainian language model fine-tuned on synthetic data to generate accurate, grounded answers. Finally, we compress the model for lightweight deployment. Evaluated under strict computational limits, our architecture demonstrates that high-quality, verifiable AI question answering can be achieved locally on resource-constrained hardware without sacrificing accuracy. Code is available at: https://github.com/trokhymovych/unlp-2026-shared-task.

1 Introduction

Large Language Models (LLMs) have emerged as universal tools, demonstrating remarkable capabilities across a wide range of natural language processing tasks (Minaee et al., 2025). While they encode vast amounts of information within their billions of parameters, this internal knowledge is strictly limited by their training data. LLMs become unreliable when tasks require highly specific knowledge - like texts with information that lie beyond models' training corpus or recent facts (Li et al., 2025). In these scenarios, they frequently fall back to so-called hallucinations to fill the gaps in their knowledge.

To bridge this gap, Retrieval-Augmented Generation (RAG) has become the definitive framework (Lewis et al., 2020). By conditioning the generative process on information retrieved from external, domain-specific databases, RAG grounds the LLM in verifiable facts, which aligns with broader efforts in automated fact-checking and knowledge verification (Thorne and Vlachos, 2018; Trokhymovych and Saez-Trumper, 2021). Implementing this, however, adds significant complexity to the data extraction and indexing pipeline. Furthermore, it causes a critical paradigm shift: the precision of the search becomes more important than the raw power of the generative model. Because LLMs rely on the provided context, retrieving incorrect information directly results in an incorrect answer (Niu et al., 2024).

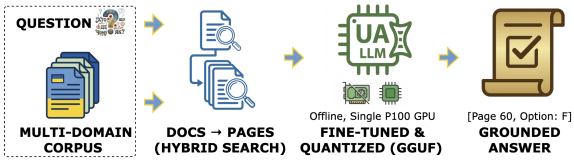

Figure 1: An end-to-end Ukrainian RAG for question answering on local deployment.

Moreover, the RAG paradigm inherently requires processing extensive retrieved contexts. At the same time, most language models are heavily optimized for English (Vargas-Parada, 2025). Applying these models to low- and medium-resource languages like Ukrainian results in a significant decrease in computational efficiency. Default tokenizers fragment Cyrillic text into significantly more subword pieces than Latin text (Maksymenko and Turuta, 2025). This exhausts memory limits, significantly slows inference, and degrades accuracy.

A recent UNLP 2026 Shared Task provided a standardized benchmark for these exact challenges, continuing the community's ongoing efforts to advance Ukrainian NLP and information integrity (Kyslyi et al., 2025; Akhynko et al., 2025). Participants were required to build a system capable of answering multiple-choice questions in Ukrainian, grounding each answer to a specific document and page within a custom, multi-domain corpus. Additionally, the organizers imposed strict

[1] https://github.com/unlp-workshop/unlp-2026-shared-task

Proceedings of the Fifth Ukrainian Natural Language Processing Conference (UNLP 2026), pages 223–229

hardware constraints: entirely offline inference on a single P100 GPU within a 9-hour limit. In practice, this meant the solution could not rely on third-party LLM providers and had to be both memory-efficient and fast.

This paper presents our solution to the UNLP 2026 Shared Task (see Figure 1), which achieved 2nd place on the final leaderboard. Our main contributions are:

1. A specialized two-stage (document and page-level) hybrid retrieval system.
2. A methodology for generating synthetic Ukrainian QA datasets for model fine-tuning.
3. A customized LLM engineered for high-speed local inference for question answering and grounding.

2 Related Work

2.1 Retrieval-Augmented Generation

Modern RAG architectures have converged on a multi-stage retrieval approach to balance latency with precision (Gao et al., 2024). These pipelines typically begin with a broad search to identify candidate documents, using fast bi-encoders (Reimers and Gurevych, 2019), sparse models such as BM25 (Robertson et al., 1994), or a combination of both. Once a candidate set is retrieved, a more computationally intensive cross-encoder reranker is applied to refine the results and ensure high relevance (Nogueira and Cho, 2019).

Recent studies suggest that hybrid retrieval, which fuses dense semantic embeddings with lexical search, consistently outperforms single methods by capturing both conceptual meaning and exact keyword matches (Akarsu et al., 2026). A common technique for this fusion is Reciprocal Rank Fusion (RRF), an unsupervised method that merges disparate ranked lists by summing the reciprocal of document ranks (Cormack et al., 2009).

Prior to retrieval, processing large documents necessitates chunking (Gao et al., 2024), a step that often causes individual segments to lose their thematic connection to the original source. To address this, contextual embeddings can be utilized to preserve document-wide intent within each fragment (Eslami et al., 2026).

2.2 Language Models for Ukrainian

Modern Large Language Models (LLMs) can effectively handle many languages (Team et al., 2025; Üstün et al., 2024), but they still favor high-resource languages, simply because they dominate the text used to train them. Similar challenges have been previously observed across various NLP tasks, from evaluating text readability to maintaining knowledge integrity and question answering (Trokhymovych et al., 2023, 2024; Zhang et al., 2023). As a result, models struggle with medium-resource languages like Ukrainian. Standard tokenizers break Ukrainian words into too many tokens (Maksymenko and Turuta, 2025). This wastes space, makes processing slower, and hurts the model's overall performance.

To address these gaps, recent work has introduced models specifically optimized for Ukrainian, such as MamayLM (Yukhymenko et al., 2025). Built on the Gemma 3 12B architecture, it underwent continual pre-training on a large, pre-filtered dataset using a combination of data mixing and model merging to gain exceptional Ukrainian cultural and linguistic proficiency. Despite its 12B-parameter size, MamayLM matches or exceeds the performance of significantly larger models, including Llama 3.1 70B, on Ukrainian-specific tasks (Yukhymenko et al., 2025).

2.3 LLM Adaptation and Deployment

LLMs are typically developed as general-purpose models capable of performing a wide range of tasks. However, they can benefit from fine-tuning for specialized applications. Given their massive size, traditional fine-tuning is often computationally impossible under hardware constraints. To address this, previous work introduced LoRA (Low-Rank Adaptation), which only updates small, low-rank matrices that serve as adapters to original model weights modifying key layers such as the attention layers (Hu et al., 2021). This approach enables the efficient adaptation of LLMs for specific tasks such as grounded question answering.

There are also several approaches that enable efficient inference under hardware constraints. In particular, we employ quantization via the GGUF format and the llama.cpp library (Gerganov, 2023). Quantization reduces the numerical precision of model weights, which significantly decreases memory usage and increases throughput with only a moderate impact on accuracy (Rajput and Sharma, 2024). The GGUF format facilitates this process by storing quantized weights alongside essential metadata, enabling efficient loading and execution across diverse hardware configurations.

3 System Architecture

This section describes our final solution for the UNLP 2026 Shared Task. Built on a RAG framework, our system employs a modular pipeline designed to preserve document structure during processing, perform relevant context retrieval across multi-domain corpora, and generate grounded answers to the questions.

3.1 Data Preparation

To convert raw PDF documents into a format suitable for Large Language Models (LLMs), we utilize `pymupdf4llm`[2] to perform layout-aware extraction. This method converts document pages into Markdown, which preserves structural elements such as tables and headers.

Only for lexical search, we implement a custom pre-processing function using `pymorphy3`[3] to perform lemmatization and tokenization. Additionally, we filter out a comprehensive list of Ukrainian stop words[4] and generic artifacts to improve the signal-to-noise ratio during the retrieval. This function is applied to both the query and the corpus for lexical search.

3.2 Hybrid Retrieval Pipeline

Our retrieval strategy is a two-stage process that first identifies the correct document and then finds the most relevant pages within that document for each question.

3.2.1 Document-Level Retrieval.

The initial search narrows the candidate documents to the single most relevant file. We employ a hybrid scoring mechanism that combines:

- Dense Retrieval: We use Perplexity embeddings (`pplx-embed-context-v1-0.6b`[5]), to capture the semantic similarity between the query and the document content. Model choice is motivated by strong performance on benchmarks and compact 0.6B size
- Sparse Retrieval: We utilize the `BM25Okapi` algorithm.

To perform the document search, we represent a query as the question concatenated with its corresponding answer options. To build the vector representation of each document, we embed its first 300 characters and use cosine similarity for ranking. For lexical search, we index each document using its full preprocessed text and use the BM25 score for ranking.

We found that for the majority of questions, the top-ranked item matches for both sparse and dense retrieval and is the correct document. For others, the correct document appears in the top two retrieved documents for at least one approach. Therefore, if the top results match, we return that document. Otherwise, we select the top two documents from each approach and use `jina-reranker-v3`[6] to find the best match. For the reranker, each document is represented by concatenating its first 300 characters with its best BM25 snippet. This algorithm allowed us to achieve near-perfect performance for document retrieval.

3.2.2 Page-Level Retrieval

Once a document is selected, we split it into small parts using syntactic chunking based on Markdown structure. Specifically, we ensure that each chunk is no longer than 500 characters, has a 10% overlap with adjacent chunks, and corresponds to only one page.

Embeddings for each chunk are calculated using the same contextual model as for document-level retrieval. Due to limited resources, we encode chunks in batches of 5 with an overlap of 2 to preserve context. To obtain the final embedding for items in the overlap, we average the results from both batches.

Embeddings are used to identify the chunks ranking via cosine similarity with the question vector. Additionally, we generate a separate ranking based on the BM25 score. We then apply Reciprocal Rank Fusion (RRF) to merge the vector and BM25 scores ranking. Finally, we use a custom cross-encoder based on `BAAI/bge-reranker-v2-m3` to rerank the top-8 candidates, providing a final list of the most relevant chunks for generation. The model was chosen for its strong performance on benchmarks and simple architecture that allows for efficient fine-tuning and inference on a P100 GPU.

[2] https://pymupdf.readthedocs.io/en/latest/pymupdf4llm/

[3] https://pypi.org/project/pymorphy3/

[4] https://github.com/skupriienko/Ukrainian-Stopwords

[5] https://huggingface.co/perplexity-ai/pplx-embed-context-v1-0.6b

[6] https://huggingface.co/jinaai/jina-reranker-v3

3.3 Answer Generation and Grounding

Based on a ranked list of chunks, we build the context out of the full text of the top-3 relevant pages. We pass the context, question, and options to the fine-tuned MamayLM-12B generative model (see the prompt in Appendix A). The model is specifically fine-tuned via Low-Rank Adaptation (LoRA) to perform the dual task of generating an answer with the page number (e.g., "A 2") to ground the response. To optimize the model for deployment in resource-constrained environments (such as Kaggle's P100 GPU), we utilize GGUF 4-bit quantization via the `llama-cpp-python` library.

4 Experimental Setup

4.1 Data

The competition provides a development dataset consisting of 461 questions and 41 documents from two distinct domains. We use this dataset as the primary data for local evaluation of our solution. It should be noted that documents in the corpus vary significantly in length, with some reaching more than 100 pages.

The Shared Task is structured as a code competition, where solution inference is performed on a hidden test set. The leaderboard is split into public (27% of the test data) and private (73% of the test data) sections, with the latter defining the final team ranking. The only information available regarding the test data is that the corpus consists of more than 240 files from a new secret domain and contains a significantly larger number of questions.

To expand the training dataset for custom reranker and foundational LLM fine-tuning, we developed an automated pipeline that synthesizes multiple-choice questions (MCQs) directly from source PDF documents. The pipeline processes documents page by page, first evaluating whether a page contains factual content suitable for question generation. It is explicitly instructed to skip non-factual sections, such as title pages, tables of contents, abbreviation lists, and general introductions.

For each page, the model generates up to ten MCQs. Each question includes exactly six answer options (A through F) with a single correct answer and reference to the document corpus reproducing the structure of the original development data. To ensure each generated question is self-contained and answerable without referencing the source document, the prompt enforces a critical constraint: every question must explicitly state the relevant entity name (e.g., the specific sport or drug).

Generation was performed using OpenAI's `gpt-4o-mini` model, configured with a temperature of 0.7. The complete prompt used for this generation is provided in Appendix A. As a result, this process yielded an additional synthetic dataset of about 7,000 questions.

4.2 Reranker Model Fine-Tuning

The custom reranker is trained using the `BAAI/bge-reranker-v2-m3` model with a hybrid loss objective that combines Listwise Cross-Entropy (weight 1.0) and Pointwise Binary Cross-Entropy (BCE) (weight 0.4) to optimize document chunk ranking. For each query, the model processes a group of candidate chunks and computes a relevance score for each. The model is trained for 5 epochs with a batch size of 2 and gradient accumulation over 4 steps. Final model selection is based on Top-1 Accuracy performance on a hold-out validation subset.

4.3 Generative Model Fine-Tuning

For the generation stage, we fine-tune the `MamayLM-Gemma-3-12B-IT-v1.0` model using Low-Rank Adaptation (LoRA) with a fixed prompt structure (see Appendix A). The training data is processed using a syntactic masking strategy, where the loss is calculated exclusively on the model's generated answer (letter and page number) by masking the prompt tokens. To accommodate long-context documents, the implementation includes a dynamic truncation mechanism with a maximum sequence length of 4000 tokens and utilizes Flash Attention 2 and gradient checkpointing for efficiency. The model is trained using the AdamW 8-bit optimizer and a cosine learning rate scheduler, with a custom evaluation suite that independently tracks accuracy for both the multiple-choice answer and the cited page number. We fine-tune the model for two epochs, first using synthetic data and subsequently on the competition development data, to ensure maximal adaptation of the model to the testing set.

5 Results

5.1 Evaluation Metrics

The evaluation metric is a weighted average that assesses three specific components of the model's output across N questions. Half of the total score

(0.5) is determined by the accuracy of the multiple-choice answers, a_i, where the model receives a point only for an exact match with the ground truth.

The remaining half of the score focuses on the retrieval reference and is split equally (0.25 each) between the document ID, d_i, and the page proximity, p_i. The document score is binary, rewarding the model for identifying the correct file. The page proximity score is more granular; it calculates the distance between the predicted and true page numbers relative to the total number of pages in the document. Crucially, this proximity credit is only awarded if the correct document was identified first ($d_i = 1$). This structure penalizes complete retrieval failures while providing partial credit for "near misses" on specific page locations within the correct document.

5.2 Competition Results

In the final ranking, we achieved second place with a score of 0.942 on the private test set and 0.920 on the public set. Our pipeline successfully processed the hidden test corpus, exactly meeting the strict 9-hour compute limit on a single P100 GPU. Local validation scores on the development set closely matched these final results, demonstrating strong generalization to the unseen domain. Finally, our two-stage hybrid retrieval achieved near-perfect accuracy for document identification and a page-level recall@3 of 0.92 on the development dataset.

6 Conclusion

We presented an efficient, end-to-end RAG system for Ukrainian QA, achieving 2nd place in the UNLP 2026 Shared Task. By combining a two-stage hybrid retrieval pipeline, fine-tuning based on synthetic data, and a quantized `MamayLM-12B model`, we demonstrated that accurate, grounded question answering is viable on local, resource-constrained hardware.

Limitations

Our approach presents several limitations, primarily related to the strict hardware constraints of the shared task. First, to minimize computational overhead, we bypass the processing of images and charts, thereby omitting visual context that could otherwise enhance retrieval and generation accuracy. Second, a marginal fraction of complex PDF pages failed during the layout extraction phase. Although Optical Character Recognition (OCR) could recover this text, the additional processing time outweighed the benefits, given the low incidence rate. Finally, we relied on standard 4-bit GGUF quantization to ensure reliable execution on the provided legacy GPU architecture (P100). Exploring alternative, state-of-the-art quantization techniques could potentially yield further performance improvements if deployed on more modern hardware.

Ethical Considerations

While our system advances Ukrainian NLP, the foundational models and synthetic data pipeline may inherit biases from their pre-training corpora. Furthermore, despite RAG's grounding mechanisms, the risk of hallucination persists. Given the inclusion of sensitive domains like pharmaceutical data, this system requires human oversight for real-world applications. On the other hand, our focus on fully offline, local inference ensures strict data privacy and reduces the environmental footprint of deployment.

We acknowledge the use of AI tools in the preparation of this manuscript. As the authors are non-native English speakers, Google Gemini and Grammarly were used to correct grammar and refine language, improving readability. Additionally, a generative AI model was utilized to create visual elements for the paper's teaser image. Ethically, these tools were not used to generate scientific claims, experimental data, or core ideas, ensuring all intellectual contributions remain solely those of the authors.

Acknowledgments

The work of Mykola Trokhymovych is supported by the Google PhD Fellowship and MCIN/AEI /10.13039/501100011033 under the Maria de Maeztu Units of Excellence Programme (CEX2021-001195-M).

References

Meftun Akarsu, Recep Kaan Karaman, and Christopher Mierbach. 2026. From BM25 to corrective RAG: Benchmarking retrieval strategies for text-and-table documents. *Preprint*, arXiv:2604.01733.

Kateryna Akhynko, Oleksandr Kosovan, and Mykola Trokhymovych. 2025. Hidden persuasion: Detecting manipulative narratives on social media during the 2022 Russian invasion of Ukraine. In *Proceedings of*

the Fourth Ukrainian Natural Language Processing Workshop (UNLP 2025), pages 194–202.

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms Condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA. Association for Computing Machinery.

Sedigheh Eslami, Maksim Gaiduk, Markus Krimmel, Louis Milliken, Bo Wang, and Denis Bykov. 2026. Diffusion-pretrained dense and contextual embeddings. *Preprint*, arXiv:2602.11151.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for Large Language Models: A survey. *arXiv preprint arXiv:2312.10997*.

Georgi Gerganov. 2023. llama.cpp: LLM inference in C/C++. `https://github.com/ggml-org/llama.cpp`. Accessed: 2026-04-06.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of Large Language Models. *CoRR*, abs/2106.09685.

Roman Kyslyi, Nataliia Romanyshyn, and Volodymyr Sydorskyi. 2025. The UNLP 2025 shared task on detecting social media manipulation. In *Proceedings UNLP 2025*, pages 105–111, Vienna, Austria (online). Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of NIPS '20*.

Moxin Li, Yong Zhao, Wenxuan Zhang, Shuaiyi Li, Wenya Xie, See-Kiong Ng, Tat-Seng Chua, and Yang Deng. 2025. Knowledge boundary of Large Language Models: A survey. In *Proceedings of ACL'25*, pages 5131–5157.

Daniil Maksymenko and Oleksii Turuta. 2025. Tokenization efficiency of current foundational large language models for the Ukrainian language. *Frontiers in Artificial Intelligence*, 8.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2025. Large Language Models: A survey. *Preprint*, arXiv:2402.06196.

Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, KaShun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2024. RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. In *Proceedings of ACL'24*, pages 10862–10878.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

Saurabhsingh Rajput and Tushar Sharma. 2024. Benchmarking emerging deep learning quantization methods for energy efficiency. In *2024 IEEE 21st International Conference on Software Architecture Companion (ICSA-C)*, pages 238–242.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Text Retrieval Conference*.

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

James Thorne and Andreas Vlachos. 2018. Automated fact checking: Task formulations, methods and future directions. In *Proceedings of COLING'18*, pages 3346–3359.

Mykola Trokhymovych, Muniza Aslam, Ai-Jou Chou, Ricardo Baeza-Yates, and Diego Saez-Trumper. 2023. Fair multilingual vandalism detection system for Wikipedia. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 4981–4990, New York, NY, USA. Association for Computing Machinery.

Mykola Trokhymovych and Diego Saez-Trumper. 2021. WikiCheck: An end-to-end open source automatic fact-checking API based on Wikipedia. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 4155–4164, New York, NY, USA. Association for Computing Machinery.

Mykola Trokhymovych, Indira Sen, and Martin Gerlach. 2024. An open multilingual system for scoring readability of Wikipedia. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6296–6311, Bangkok, Thailand. Association for Computational Linguistics.

Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. 2024. Aya model: An instruction finetuned open-access multilingual language model. In *Proceedings of ACL'24*, pages 15894–15939.

Laura Vargas-Parada. 2025. Large language models are biased — local initiatives are fighting for change. *Nature*.

Hanna Yukhymenko, Anton Alexandrov, and Martin Vechev. 2025. MamayLM: An efficient state-of-the-art Ukrainian LLM. https://huggingface.co/blog/INSAIT-Institute/mamaylm.

Xiang Zhang, Senyu Li, Bradley Hauer, Ning Shi, and Grzegorz Kondrak. 2023. Don't trust ChatGPT when your question is not in English: A study of multilingual abilities and types of LLMs. In *Proceedings of EMNLP'23*, pages 7915–7927.

A Prompts

A.1 Answer Generation Prompt

The following prompt is used for final answer generation with page grounding (Figure 2).

```
Context (excerpts from PDF files - each excerpt
    is
separated by ``` characters and contains a page
number enclosed in []):
```
Page: [<page_number_1>]
<page_text_1>
```
```
Page: [<page_number_2>]
<page_text_2>
```
```
Page: [<page_number_3>]
<page_text_3>
```

Question: <question>
Options:
A: <option_A>
B: <option_B>
C: <option_C>
D: <option_D>
E: <option_E>
F: <option_F>
Instructions:
- Answer the Question using the Context.
- Return the letter of the correct answer (A B C
    D E F)
  and the page number where the information was
    found,
  separated by a space (e.g., A 1).
- Think carefully; first eliminate the obviously
  irrelevant options.
```

Figure 2: Prompt template for answer generation with grounding (translated from Ukrainian).

A.2 Synthetic Question Generation Prompt

The following prompt is used for synthetic MCQ generation (Figure 3).

```
SYSTEM:
You are an expert Ukrainian-language exam
    question writer.

DOMAIN CONTEXT:
{domain_description}

YOUR TASK:
You will receive the text of a single page from
    a Ukrainian PDF document. You must:

1. IDENTIFY the specific subject: the exact
    sport name
   (e.g. "strongman", "sambo") or drug name
   (e.g. "retabolil", "fervex"). This is the
    ENTITY NAME.
2. DECIDE whether this page contains specific
    factual
   content suitable for question generation (
    specific
   rules, dosages, penalties, contraindications,
     etc.).
   SKIP pages that are tables of contents, title
     pages,
   abbreviation lists, or general introductions.
3. If suitable, generate up to 10 MCQs that:
   - Are written entirely in Ukrainian.
   - CRITICAL: Every question MUST explicitly
    name the
     ENTITY NAME. Generic questions are
     FORBIDDEN.
   - Are answerable ONLY from the provided page
    text.
   - Have exactly 6 options (A-F), one correct
    answer,
     and 5 plausible distractors.
   - NEVER use quotation marks or braces.
   - Match the style of: {few_shot_examples}
4. If NOT suitable, return an empty questions
    list.

RESPONSE FORMAT (strict JSON):
{
  "entity_name": "<sport or drug name>",
  "questions": [
    {
      "question": "...",
      "A": "...", "B": "...", "C": "...",
      "D": "...", "E": "...", "F": "...",
      "correct_answer": "A"
    }
  ]
}

Return ONLY valid JSON. "correct_answer" must be
     one of: A, B, C, D, E, F.
If not suitable: {"entity_name": "", "questions":
     []}

USER:
Domain: {domain}
Document page text: {page_text}
```

Figure 3: Prompt template used for synthetic question generation. Note: examples appear in Ukrainian in the actual prompt.