TextSentinels @ FinMMEval 2026 Task 1: A Multilingual Routed Retrieval-Augmented System for Financial Multiple Choice Questions

Durairaj Thenmozhi[1], Amrit Gopinath[2], Adithya Sivakumar[2,*], Amudhan A[2] and Aakash Balasubramanian[2]

[1]*Shiv Nadar University, Chennai, India*
[2]*Sri Sivasubramaniya Nadar College of Engineering, Chennai, India*

Abstract

We present a multilingual routed retrieval-augmented system for FinMMEval 2026 Task 1, a financial multiple-choice question answering benchmark covering English, Chinese, and Arabic. Our approach builds a shared FAISS retrieval index from the official public Task 1 Hugging Face release after normalizing heterogeneous dataset schemas into a unified multiple-choice format. The retrieval corpus is constructed from public sources, including `SahmBenchmark/arabic-accounting-mcq_train`, `SahmBenchmark/arabic-business-mcq_training_standardized`, `TheFinAI/flare-es-multifin`, `Tomas08119993/finmmeval-cfa-cpa`, and `TheFinAI/plutus-multifin`. During inference, a router implemented with `qwen/qwen3-32b` classifies each question into factual, reasoning-heavy, or exhibit-best-effort modes. Answer options are then scored by a two-model ensemble using `qwen/qwen3-32b` and `openai/gpt-oss-20b` through the Groq OpenAI-compatible API, and the two score distributions are merged using fixed, route-dependent weights. The final system balances multilingual coverage, retrieval quality, and inference stability. It was able to achieve competitive development and final-test results across the selected language tracks.

Keywords

Financial QA, Multilingual NLP, Retrieval-Augmented Generation, FAISS, Option Scoring, FinMMEval

1. Introduction

The FinMMEval 2026 Task 1 focuses on testing multilingual financial multiple-choice question answering [1, 2]. This task is quite difficult to perform due to the inclusion of many areas that require special expertise, such as valuation, accounting, ethics, corporate finance, and regulation. Multilingualism increases complexity, as the same reasoning approach does not apply uniformly across different languages and types of questions.

In our final constructed system, we have ensured that the system works for English, Chinese, and Arabic. We chose these three tracks as we were able to validate a retrieval-based pipeline only for these three tracks during development. We did not proceed with unconstrained answer generation. Instead, we opted for a routed option-wise scoring ensemble model. Our retrieval setup was inspired by retrieval-augmented generation [3], while our prompt design was informed by work on chain-of-thought reasoning [4]. We have a multilingual FAISS index that is based on multiple choices from public Task 1 data, along with a classification model (router) that decides whether a particular question is classified as factual, reasoning, or best-effort. This is done using `qwen/qwen3-32b`. The final answer is obtained by scoring options with `qwen/qwen3-32b` and `openai/gpt-oss-20b`, then combining the score distributions with route-dependent weights.

The fundamental premise of our system design lies in the heterogeneity of the multilingual finance MCQ question answering task [5, 6]. Some questions can be efficiently answered by using semantic searches to find relevant solutions, while others rely heavily on short reasoning chains or even guesswork

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany
*Corresponding author.
thenmozhid@snuchennai.edu.in (D. Thenmozhi); amrit2410182@ssn.edu.in (A. Gopinath); adithya2410402@ssn.edu.in (A. Sivakumar); amudhan2410622@ssn.edu.in (A. A); aakash2410407@ssn.edu.in (A. Balasubramanian)


based on the visible context. We thus decided to break down the process into distinct operational modes and use a unique method of inference for each, without complicating the workflow.

Our final system consists of three elements: a multilingual retrieval vector index built from the public Task 1 Hugging Face data sources, a router-model based question classification, and option scoring and answering with a two-model ensemble. We also explored stronger models, complex routing, and ensembles, but these variants did not consistently outperform the simple routed system. The final system design balances multilingual coverage, retrieval quality, and inference stability and performs consistently on the leaderboard.

2. System Description

Our final system is a multilingual, routed retrieval-augmented multiple-choice inference pipeline. The system has four stages: public data normalization, FAISS index construction, route prediction, and route-aware option scoring.

2.1. Public data sources

The retrieval dataset was built exclusively from the public Task 1 Hugging Face sources provided by the organizers. The public sources used in the final system included the following datasets and splits:

- `SahmBenchmark/arabic-accounting-mcq_train`, split `train`,
- `SahmBenchmark/arabic-business-mcq_training_standardized`, split `train`,
- `TheFinAI/flare-es-multifin`, split `test`,
- `Tomas08119993/finmmeval-cfa-cpa`, split `train`,
- `TheFinAI/plutus-multifin`, split `train`.

These sources cover Arabic, Chinese, English, Spanish, and Greek financial multiple-choice data. Following the organizers' citation guidance for Task 1, we cite the underlying source papers for the corpora we drew on: SAHM for the Arabic `SahmBenchmark` sources [7], and RealFin for the English and Chinese portions of the release [8]. Language for each row was assigned from a per-row `language` field released with the public data (e.g., rows in `plutus-multifin` tagged `zh` or `chinese`) rather than inferred from the dataset name itself; a small number of rows with missing or ambiguous language tags were assigned a language with lightweight script- and keyword-based heuristics as a fallback. Because the public corpus is multilingual, we embedded examples with a multilingual sentence encoder and built a single shared FAISS index for retrieval. Organizer-held development questions were not inserted back into the retrieval corpus, and final-test labels were not available to the system.

2.2. Data normalization

The public Task 1 datasets were not organized in a similar manner. Different datasets used different split names, column names, answer formats, option layouts, and question styles. In some cases, the answer choices were given as separate fields, while in others, they were mixed into longer text fields. Some datasets used letter labels such as A/B/C/D, while others used numbers or other answer metadata. One source also needed direct Parquet loading, as the standard dataset loader was unable to interpret it properly.

To make these datasets usable for retrieval, we converted all valid rows into one shared multiple-choice format. Each example was stored with a row ID, dataset name, split, language, normalized question, normalized answer options, and answer label when available. This step was important, as the rest of the system expects every example to look like a regular multiple-choice question.

2.3. FAISS store construction

After normalization, every public example was converted into a passage-style text representation for embedding. Rather than storing generic documents, the code converts each row into a passage-style MCQ text containing the dataset identifier, language, question, options, and, when available, the answer or reason.

We embedded the resulting texts using the multilingual sentence-transformer model `sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2` [6], in batches of 32. The embeddings were L2-normalized and stored in a FAISS index for nearest-neighbor search [9]. Metadata for each vector entry was stored separately in JSONL form so that each retrieved vector could be mapped back to its normalized MCQ record at inference time.

This design makes the retrieval component strictly example-based. When the system sees a new target question, it searches for semantically similar solved public MCQ instances rather than arbitrary financial documents [3]. In our experiments, this was particularly useful for conceptual finance, accounting, and regulation-style questions, where similar question patterns recur across the public release.

2.4. Question routing

The second major component of the system is route prediction. In early stages of development, we observed that a one prompt template is suboptimal for all Task 1 items. Some questions are short and factual, some require multi-step reasoning, and some refer to missing exhibits or long passages in ways that can mislead a retrieval-heavy prompt.

To address this, we used `qwen/qwen3-32b` as a lightweight router. For each target question, the router predicts one of three categories:

- **factual_rag**: factual or conceptual questions where retrieval of similar solved examples is likely to help;
- **reasoning_math**: questions that appear calculation-oriented or require multi-step financial reasoning;
- **exhibit_best_effort**: questions whose wording suggests dependence on partially missing exhibit, table, or passage context.

The router also predicts a coarse difficulty label, though in the final submitted system, the route decision was more important than the difficulty tag itself. This decomposition allowed us to assign different retrieval depths and different answer-scoring behavior to different question types. In the final submitted system, factual questions retrieve two examples, reasoning-heavy questions retrieve one example, and exhibit-style questions use no retrieved examples.

The router receives a system message of *"You are a routing classifier for financial MCQ solving. Return strict JSON only."* together with a user prompt built from the following template:

```
Classify this financial multiple-choice question for routing.
Return JSON only with keys: route, difficulty, rationale_short.
route must be one of: "factual_rag", "reasoning_math", "exhibit_best_effort".
difficulty must be one of: "easy", "medium", "hard".

Language: <language>
Question:
<question>
Options:
<A>. <option A>
<B>. <option B>
...
```

The router is queried with a strict JSON response format; if the endpoint rejects the JSON schema for a given request, the system retries once without the schema constraint and falls back to regex-based JSON extraction from the raw response text.

2.5. Route-aware option scoring

The final answer selection stage uses a routed option-scoring ensemble served through Groq's OpenAI-compatible API. In the final system, Qwen serves dually as the router and as the factual/conceptual scorer, while GPT-OSS-20B serves as the reasoning-oriented scorer.

Instead of asking the models to generate a final answer directly, we formulate the task as option scoring. For each target question, the system constructs a prompt containing the visible question, the candidate options, route-specific instructions, and a small retrieved support set from the FAISS index based on the designated retrieval depth. Both models then assign scores to the available answer choices.

The scoring system message is *"You are a careful financial multiple-choice solver. Reason privately. Return strict JSON only."*, and the user prompt follows this template:

```
Score each option for correctness.
Return JSON only with keys: scores, best_answer.
scores must map each option letter to a non-negative number.
best_answer must be one option letter.

<route instruction>

Retrieved examples:
Example 1 (similarity=<score>)
Question: <retrieved question>
<A>. <option A>
...
Correct answer: <retrieved answer>
...  (repeated for each retrieved example, or "Retrieved examples: none")

Target language: <language>
Question:
<question>
Options:
<A>. <option A>
...
```

The `<route instruction>` depends on the predicted route:

- **factual_rag**: *"This is likely factual or conceptual. Use retrieved examples as soft reference patterns, not as direct evidence."*
- **reasoning_math**: *"This is likely calculation-heavy or multi-step. Reason privately, use retrieval only if directly relevant, and score each option."*
- **exhibit_best_effort**: *"This may reference an exhibit or passage. If some context seems missing, infer from the visible question and options anyway."*

Qwen is queried with a plain JSON response format. GPT-OSS-20B is queried with a strict JSON schema constraining `scores` to one non-negative number per option label and `best_answer` to one of the option letters, and with an explicit reasoning-effort parameter set to `medium` for easy/medium-difficulty questions and `high` for questions the router labeled `hard`. Neither model call sets an explicit sampling temperature, so provider defaults apply.

Each model's raw scores are clipped to be non-negative and re-normalized to sum to 1; if all scores are non-positive or missing, the distribution falls back to a uniform distribution over the available option labels. The two normalized distributions are then combined into a single merged distribution using fixed, route-dependent weights w_{qwen} and $w_{\text{gpt-oss}}$, shown in Table 1, and the option with the highest merged score is emitted as the final answer.

The route-dependent weights were set manually to reflect each model's designated role per route (e.g., upweighting the reasoning-oriented scorer on `reasoning_math`, where it also receives escalated reasoning effort) rather than tuned via a hyperparameter search on held-out data; exploring learned or validation-tuned weights is left to future work.

Table 1
Route-dependent merge weights applied to the two normalized score distributions.

Route	w_{qwen}	$w_{\text{gpt-oss}}$
factual_rag	0.55	0.45
reasoning_math	0.35	0.65
exhibit_best_effort	0.60	0.40

This option-scoring formulation had two practical benefits. First, it improved output stability and reduced malformed responses compared to direct answer generation. Second, it made ensembling more straightforward, since each model contributes a score distribution rather than a single, possibly brittle generated token.

2.6. Implementation details

During development, we also evaluated alternative variants using `gpt-4o-mini`, `llama-3.3-70b-versatile`, and `openai/gpt-oss-120b`, but these were not part of the final submitted configuration. The router and scoring models were `qwen/qwen3-32b` and `openai/gpt-oss-20b`. Retrieval depth was route-dependent, with top-$k = 2$ for factual questions, top-$k = 1$ for reasoning-heavy questions, and no retrieval for exhibit-best-effort cases in the tuned setup. Embeddings were computed in batches of 32 and L2-normalized before indexing in FAISS. We normalized the public datasets into a shared multiple-choice schema before embedding. Neither router nor scorer calls set an explicit decoding temperature, so provider (Groq) default sampling was used throughout. We submitted results for English, Chinese, and Arabic.

For reproducibility, the pipeline was implemented in Python with pinned minimum dependency versions: `datasets` $\geq$ 3.0.0, `faiss-cpu` $\geq$ 1.8.0, `huggingface_hub` $\geq$ 0.32.0, `numpy` $\geq$ 1.26.0, `openai` $\geq$ 1.82.0 (used as the OpenAI-compatible client for the Groq API), `pyarrow` $\geq$ 18.0.0, and `sentence-transformers` $\geq$ 3.0.0. All embedding and FAISS indexing ran on `faiss-cpu` on a standard CPU host; no GPU was required locally, since router and scorer inference was delegated to the hosted Groq API rather than run on local hardware.

3. Experimental Development

The final Task 1 system was developed through several rounds of iteration over retrieval, prompting, routing, and model combination. We did not arrive at the final routed ensemble directly. Instead, we first tested simpler retrieval-based systems and then compared several routed variants before selecting the final submission model.

3.1. Early Retrieval and Prompting Baselines

Our earliest experiments used direct retrieval-augmented prompting. These systems retrieved similar public multiple-choice examples and passed them to a single answering model, `gpt-4o-mini`, together with the target question. They were useful mainly for exposing practical issues in the public data, such as inconsistent option formats, mixed-language examples, and long scenario questions that needed stronger normalization.

After moving to a persistent multilingual FAISS index built from normalized public Task 1 resources, retrieval became much more stable across English, Chinese, and Arabic. This was an important improvement, because it gave us a shared retrieval layer that all later systems could use.

We also tested several prompt styles, including direct answer generation, short hidden reasoning instructions, retrieval-heavy prompts, and stricter answer-only formats. A clear pattern emerged: stronger output control helped format stability, but too much retrieved context often hurt answer quality. In practice, one or two highly relevant examples were usually better than many loosely related

examples. The prompt-refined baseline kept `gpt-4o-mini` as the answering model but changed the prompt structure and output constraints.

3.2. Alternative Systems Explored and Their Analysis

Note on evaluation sets. The following ablation (Table 2) and the organizer leaderboard results reported later (Table 3) are two distinct evaluation sets and are not directly comparable. Table 2 uses a fixed, mixed-language, 50-question internal development slice constructed by us purely to compare candidate architectures against one another. Table 3 uses the organizers' own per-language development and final-test leaderboards, which are the basis for our actual submissions.

For reproducibility, all alternative architectures were evaluated on the same fixed 50-question development slice. This slice was created automatically from the public Task 1 target pool by combining examples from `SahmBenchmark/arabic-accounting-mcq_eval` (split `test`), `SahmBenchmark/arabic-business-mcq_eval` (split `test`), `TheFinAI/plutus-multifin` (split `validation`), and `Tomas08119993/finmmeval-cfa-cpa` (split `train`). We filtered this pool to English, Chinese, Arabic, and Greek, applied a deterministic shuffle with random seed 42, and selected the first 50 questions.

The sampled 50-question target slice was drawn from public Task 1 sources that partially overlapped with the retrieval corpus at the dataset level. Exact sampled target items were excluded from retrieval by (dataset, row) identifier during evaluation, but this exclusion operates only at the level of exact identifiers: we did not apply any further near-duplicate or embedding-similarity-based deduplication between the 50-question slice and the retrieval corpus. As a result, items that are semantically similar to, but not identical with, a given target question may still be retrievable, and the 72.0% figure reported below should be read as an optimistic upper bound on genuinely held-out development accuracy for this internal slice, rather than a fully leakage-free estimate.

Table 2 summarizes the main development results.

Table 2
Architecture-selection results on the fixed 50-question internal development slice (mixed languages; used only to compare candidate architectures, not an organizer leaderboard).

Architecture	Main model(s)	Correct / 50	Accuracy
Basic multilingual FAISS RAG baseline	`gpt-4o-mini`	31 / 50	62.0%
Prompt-refined FAISS RAG baseline	`gpt-4o-mini`	29 / 50	58.0%
Routed weighted ensemble variant	`qwen/qwen3-32b`, `llama-3.3-70b-versatile`, `openai/gpt-oss-20b`	33 / 50	66.0%
Route-aware dual scorer with filtered retrieval	`qwen/qwen3-32b`, `openai/gpt-oss-20b`, `openai/gpt-oss-120b`	36 / 50	72.0%
Final routed ensemble system	`qwen/qwen3-32b`, `openai/gpt-oss-20b`	36 / 50	72.0%

The basic multilingual FAISS RAG baseline, which used `gpt-4o-mini` as a single answering model over retrieved examples, reached 62.0% accuracy on this internal slice. This confirmed that retrieval over solved financial MCQ examples was useful, especially for factual and concept-matching questions. However, it was weaker on questions that required arithmetic reasoning, multi-step inference, or best-effort handling of incomplete exhibits.

The prompt-refined FAISS RAG baseline, which still used `gpt-4o-mini` but with a more controlled prompt and stricter answer formatting, reduced accuracy to 58.0%. In this version, the prompt gave stronger instructions about using retrieved examples and returning a clean answer format. Although this improved output formatting, it also made the system more brittle when retrieval was only partially

relevant. When the retrieved examples were topically similar but not structurally similar, the model sometimes followed the wrong pattern too closely.

We then tested a routed weighted ensemble variant, which reached 66.0% accuracy. In code, this system used `qwen/qwen3-32b` as a router, `llama-3.3-70b-versatile` as the factual scorer, and `openai/gpt-oss-20b` as the reasoning scorer. The router assigned each question to one of three routes: `factual_rag`, `reasoning_math`, or `exhibit_best_effort`. Retrieval depth depended on the route, and the two scorers were combined using fixed route-specific weights. This was a useful step forward because it showed that explicit route-based decomposition improved over plain retrieval baselines. However, it still did not reach the best score.

Another important variant was a route-aware dual scorer with filtered retrieval, which reached 72.0% accuracy. This system again used `qwen/qwen3-32b` as the router, but changed the downstream scoring strategy. For factual routes, it paired `qwen/qwen3-32b` with `openai/gpt-oss-20b`; for reasoning routes, it paired `openai/gpt-oss-20b` with `openai/gpt-oss-120b`; and for exhibit-style questions, it used only `openai/gpt-oss-20b`. It also introduced route-specific similarity thresholds to filter weak retrieval matches, and effectively disabled retrieval for exhibit-style questions. In practice, this design helped because it reduced noisy retrieval and matched stronger models to harder routes more carefully.

At the same time, this variant was more complex than our final submission design. Although it matched the best development accuracy, it introduced additional model coordination and higher inference cost. More importantly, many remaining failures were still caused by weak retrieval evidence, ambiguous multilingual phrasing, or missing exhibit information. Changing the scoring combination did not consistently solve those deeper issues.

These experiments also clarified the main failure modes in Task 1. First, retrieval mismatch remained a major problem: semantically similar neighbors were not always good teaching examples for the target question. Second, stronger prompts could sometimes amplify retrieval errors instead of fixing them. Third, route-based systems depended on both correct routing and good route-specific retrieval settings. Finally, exhibit-based questions remained difficult when the available text did not capture the missing figure, chart, or passage.

3.3. Why We Selected the Final Architecture

These development results motivated our final choice. The final routed ensemble matched the strongest observed development accuracy at 72.0% on the internal slice, while remaining simpler and easier to analyze than the more complex route-aware dual-scorer variant. It preserved the main benefit of route-based decomposition without adding unnecessary model coordination. For this reason, we selected it as the final Task 1 submission architecture.

4. Results and Discussion

We evaluated our system on the organizer-provided language-specific development leaderboards for English, Chinese, and Arabic. To execute the final submissions, the core pipeline was wrapped in language-specific driver scripts that read the organizer's public files and templates directly. As noted above, the development and final-accuracy figures below come from the organizers' own per-language leaderboards, distinct from the fixed 50-question internal slice used for architecture selection in Table 2.

Table 3 summarizes the development and final-test results.

On the final test set, the system placed 6th in English, 6th in Chinese, and 7th in Arabic, putting it in the middle of all participating systems across the selected languages.

4.1. Discussion

The results suggest that the main gains in Task 1 came from route-aware inference, rather than simply increasing the model size or retrieval volume. The multilingual FAISS index was effective as a shared

Table 3
Organizer leaderboard development and final-test results for the three submitted language tracks.

Language	Dev correct/total	Dev accuracy	Final correct/total	Final accuracy
English	59/70	84.29%	163/200	81.50%
Chinese	53/71	74.65%	153/200	76.50%
Arabic	83/97	85.57%	172/200	86.00%

retrieval backbone, but its usefulness relied heavily on careful normalization and conservative retrieval depth. In practice, the system performed best when factual, reasoning-heavy, and exhibit-dependent questions were handled separately rather than through a single uniform prompting strategy.

We also observed that option scoring was more reliable as compared to unconstrained answer generation. Constraining the models to score answer choices improved output stability and made ensembling easier to control. This was important in a benchmark setting, where malformed answers can directly reduce effective system performance even when the underlying reasoning is partly correct.

At the same time, several negative results were informative. Stronger factual substitutions, heavier retrieval, and more complex multi-model variants did not consistently improve development accuracy. In many cases, they added latency and coordination overhead without solving the core failure modes, which were usually retrieval mismatch, ambiguous multilingual phrasing, or missing exhibit context. Overall, the final routed ensemble provided the best balance between accuracy, simplicity, and inference stability.

Finally, the results across the three submitted languages suggest that the public Task 1 release already contains enough multilingual structure to support a competitive shared retrieval backbone. Although language-specific optimization remains possible, our experiments indicate that a unified multilingual pipeline can still perform well when combined with careful normalization and route-aware inference.

5. Conclusion

We presented a multilingual routed retrieval-augmented system for FinMMEval 2026 Task 1 and applied it to English, Chinese, and Arabic. We built a multilingual FAISS index from normalized public Task 1 multiple-choice examples and used a router to classify each target question into factual, reasoning-heavy, or exhibit-best-effort modes. The final answer was obtained by scoring options with `qwen/qwen3-32b` and `openai/gpt-oss-20b`, combining the score distributions with route-dependent weights.

Our experiments showed that the most effective configuration was not the most complex one. A restrained two-model routed ensemble with route-specific retrieval depth outperformed heavier experimental multi-model variants and stronger but more cumbersome factual substitutions. On the selected language tracks, the final system achieved strong development performance and competitive final-test rankings.

There are several natural directions for future work. One is to improve handling of exhibit-dependent questions, especially when crucial tabular or passage context is only partially available. Another is to explore cleaner language- or domain-partitioned retrieval stores without sacrificing the benefits of shared multilingual embeddings, and to apply embedding-similarity-based deduplication between development slices and the retrieval corpus rather than relying solely on exact-identifier exclusion. Finally, lightweight supervised fine-tuning on the public Task 1 collection may offer additional gains beyond the prompt-and-retrieval regime explored in this submission.

Acknowledgments

We thank the organizers of FinMMEval 2026 for releasing a multilingual public dataset collection and language-specific leaderboard infrastructure for Task 1.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT 5 for language editing, grammar correction, and drafting assistance. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] Z. Xie, Y. Dai, R. Elbadry, V. Jani, X. Peng, L. Qian, G. Georgiev, D. Dimitrov, F. Zhang, J. Huang, J. Geng, Y. Chen, Y. Yuan, H. Wu, Y. Wang, I. Koychev, V. Stoyanov, M. Song, Y. Chen, S. Liu, P. Nakov, Overview of FinMMEval 2026: Multilingual and multimodal financial evaluation, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction, Proceedings of the Seventeenth International Conference of the CLEF Association (CLEF 2026), Springer Lecture Notes in Computer Science LNCS, Jena, Germany, 2026.

[2] Z. Xie, Y. Dai, R. Elbadry, V. Jani, G. Georgiev, D. Dimitrov, F. Zhang, X. Peng, L. Qian, J. Huang, J. Geng, Y. Chen, Y. Yuan, H. Wu, Y. Wang, I. Koychev, V. Stoyanov, M. Song, Y. Chen, S. Liu, P. Nakov, Overview of the FinMMEval 2026 task 1: Multilingual financial multiple-choice question answering, in: CLEF 2026 Working Notes, CEUR Workshop Proceedings, CEUR-WS.org, Jena, Germany, 2026.

[3] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive NLP tasks, in: Advances in Neural Information Processing Systems, volume 33, 2020, pp. 9459–9474. URL: https://proceedings.neurips.cc/paper_files/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

[4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. V. Le, D. Zhou, Chain-of-thought prompting elicits reasoning in large language models, in: Advances in Neural Information Processing Systems, volume 35, 2022, pp. 24824–24837. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract.html.

[5] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W.-t. Yih, Dense passage retrieval for open-domain question answering, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 6769–6781. URL: https://aclanthology.org/2020.emnlp-main.550/. doi:10.18653/v1/2020.emnlp-main.550.

[6] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using Siamese BERT-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3982–3992. URL: https://aclanthology.org/D19-1410/. doi:10.18653/v1/D19-1410.

[7] R. Elbadry, S. Ahmad, A. Heakl, D. Bouch, M. Ahsan, M. AlMahri, M. E. khalil, Y. Wang, S. Lahlou, S. Ananiadou, V. Stoyanov, J. Huang, X. Peng, P. Nakov, Z. Xie, SAHM: A benchmark for arabic financial and shari'ah-compliant reasoning, 2026. URL: https://arxiv.org/abs/2604.19098. doi:10.48550/arXiv.2604.19098. arXiv:2604.19098.

[8] Y. Dai, Y. Lin, Z. Xie, Y. Wang, RealFin: How well do LLMs reason about finance when users leave things unsaid?, 2026. URL: https://arxiv.org/abs/2602.07096. doi:10.48550/arXiv.2602.07096. arXiv:2602.07096.

[9] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, IEEE Transactions on Big Data 7 (2021) 535–547. URL: https://doi.org/10.1109/TBDATA.2019.2921572. doi:10.1109/TBDATA.2019.2921572.