RMC at BioASQ 2026: A local RAG approach for end-to-end Biomedical QA

Notebook for the BioASQ Lab at CLEF 2026

Daniel Sims[1], Kassandra Williams[1] and Sam Henry[2,*]

[1]Christopher Newport University, 1 Avenue of the Arts, Newport News, Virginia, 23606, USA
[2]Randolph Macon College, 114 College Ave, Ashland, Virginia, 23005, USA

Abstract

This paper describes our submissions to the 2026 BioASQ challenge. We competed in Task 14b Phases A, A+, and B, and the Synergy task. Given a biomedical question, these tasks require systems to retrieve relevant articles and text snippets within those articles and to generate answers. Our approach is based on a local retrieval-augmented generation (RAG) pipeline that combines dense vector retrieval with a 13B parameter large language model for answer generation.

The system consists of three primary stages: document retrieval, snippet extraction and ranking, and answer generation. For retrieval, we constructed PubMed vector databases using `nomic-embed-text-v1.5` and performed K-nearest neighbors similarity search using Facebook AI Similarity Search (Faiss). We evaluated both document-level and sentence-level embedding strategies. For snippet extraction, we compared a sentence splitting approach against a BERT-based span detection model. Candidate snippets were reranked using the cosine similarity between question and snippet embeddings. For answer generation, we used `Synthia-13B-GPTQ` and evaluated alternative prompting strategies, including using only the top-ranked context versus concatenating multiple retrieved contexts. Our experiments demonstrate that locally deployable RAG systems can achieve competitive performance on biomedical question answering tasks.

Keywords

Biomedical Question Answering, Information Retrieval, Retrieval Augmented Generation

1. Introduction

In this paper, we describe our submissions to the 2026 BioASQ challenge [1] which include systems for BioASQ 2026 Synergy [2] and BioASQ Task 14b [2] Phases A, A+, and B. Given a question, these tasks require systems to retrieve relevant biomedical articles and snippets (Phase A and Synergy) and generate answers (Phase A+, Phase B, and Synergy). Our approach is a locally run retrieval-augmented generation (RAG) [3] system that combines dense vector retrieval with a 13-billion-parameter (13B) large language model (LLM) for answer generation. Across our submissions, we explored different strategies for document retrieval, snippet extraction, and context incorporation.

Our system consists of three primary stages: (1) document retrieval, (2) snippet extraction and ranking, and (3) answer generation. For document retrieval, we constructed a PubMed vector database using Nomic AI's `nomic-embed-text-v1.5` [4] and performed similarity search using Facebook AI Similarity Search (Faiss) [5]. We experimented with both document embeddings and sentence embeddings to evaluate the impact of embedding granularity.

For snippet extraction and ranking, we compared two approaches. The first was a sentence-splitting approach using the Stanza [6] GENIA sentence segmentation model. The second employed a BERT-based [7] span detection model to identify text spans from retrieved abstracts. For both methods we ranked the snippets using cosine similarity between snippet embeddings and the question embedding generated by `nomic-embed-text-v1.5`.

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany
*Corresponding author.
daniel.sims@cnu.edu (D. Sims); kassandra.williams@cnu.edu (K. Williams); samuelhenry@rmc.edu (S. Henry)
0000-0002-7246-1151 (S. Henry)

The answer generation stage used `Synthia-13B-GPTQ` [8], a LLaMA 2-based [9] model. Critical to this system are prompting and output-processing strategies. We experimented with different methods for incorporating retrieved contexts into prompts. Specifically, we compared using only the top-ranked context against concatenating all retrieved contexts prior to generation.

Our system achieved competitive results across the BioASQ tasks. In the remainder of this paper, we present the background and motivation for our approach, describe each submitted run, present results, and discuss conclusions and future work.

2. Background

2.1. Biomedical Question Answering

Question Answering (QA) is a field of natural language processing (NLP) dedicated to creating systems that automatically answer questions posed to them in the form of natural human language. Biomedical QA is a subdomain of the general QA field which focuses on answering biomedical questions. Biomedical QA has application such as serving as diagnosis aids for physicians [10] and allowing consumers to better inform themselves about their health [11].

Studies on evidence based medicine [12] have shown that Biomedical QA is a particularly difficult problem not only due to the complexity of natural language, but also because biomedical literature is difficult for most of the general public to understand. Russel-Rose and Chamberlain [13] estimate that it takes a biomedical expert around four hours to answer medical questions utilizing existing biomedical search tools and databases. These search tools typically focus on information retrieval rather than QA, and return full articles rather than succinct answers. It is the goal of Biomedical QA to build better tools for biomedical researchers, physicians, and the general public.

2.2. BioASQ

BioASQ [14][1] is an annual biomedical NLP challenge focused on semantic indexing and biomedical QA. One of its primary tracks, Task 14b, evaluates end-to-end biomedical QA systems through multiple subtasks centered on information retrieval and answer generation.

Task 14b is divided into several phases. Phase A focuses on information retrieval, where participants are given biomedical questions and must retrieve relevant PubMed [15] articles and supporting text snippets. Phase B focuses on answer generation: given a question and human-annotated supporting snippets, systems must generate answers. BioASQ also includes Phase A+, which combines retrieval and generation into a single end-to-end task. The Synergy task is also an end-to-end task focused on iterative refinement.

Questions in BioASQ belong to one of four categories: `yes/no`, `factoid`, `list`, and `summary`. `Yes/no` questions require binary "yes" or "no" answers, `factoid` questions require a single term, `list` questions require multiple terms, and `summary` questions require short paragraph-length explanatory responses. In addition, all question types (excluding summary) also include `ideal` answers which, like summary questions are short paragraph-length responses.

To support these tasks, BioASQ releases datasets containing multiple annotations, including: `body` (question text), `type` (question category), `documents` (relevant PubMed article URLs), `snippets` (relevant text passages), `exact answer` (short-form answer), and `ideal answer` (long-form explanatory answer). Each year, the previous test set is incorporated into the training data and a new test set is released for the current competition [16]. Competition batches are released incrementally, and participants are typically given 24 hours to submit system predictions for each batch.

Phase	Run	Embedding Type	Snippet Extraction	Snippets Used
Synergy	RMC_1	document	span detection	first
	RMC_2	document	span detection	all
	RMC_3	sentence	span detection	first
	RMC_4	sentence	span detection	all
Phase A	RMC_1	sentence	span detection	-
	RMC_2	sentence	sentence splitter	-
	RMC_3	sentence	combined	-
	RMC_4	document	span detection	-
	RMC_5	document	combined	-
Phase A+	RMC_1	sentence	span detection	first
	RMC_2	sentence	sentence splitter	first
	RMC_3	sentence	combined	first
	RMC_4	sentence	span detection	all
	RMC_5	sentence	sentence splitter	all
Phase B	RMC_1	-	gold	first
	RMC_2	-	gold	all

Table 1
Descriptions of experimental runs across phases.

3. Methodology

Our system is based on RAG framework and is divided into two primary stages: information retrieval and answer generation. The information retrieval stage retrieves relevant contexts (documents and snippets), while the answer generation stage produces answers using the retrieved contexts (Phase A+, Synergy) or gold-standard human annotated contexts (Phase B). Figures 1 and 2 outline each primary component. Detailed descriptions of each component are provided in subsequent subsections.

During system development, we evaluated multiple approaches for several components of the pipeline. These alternative methods, shown as adjacent boxes in Figures 1 and 2, include:

1. **Document Retrieval** — retrieval of relevant PubMed PMIDs and abstracts. We compared document embeddings against sentence embeddings.
2. **Snippet Extraction** — extraction of relevant passages from the retrieved documents. We compared a sentence-splitting method against a BERT-based span detection model.
3. **Answer Generation** — generation of answers using the evidence. We compared using only the highest-ranked snippet as context against concatenating all retrieved snippets into a single context string.

These experiments correspond to our submitted runs, which are summarized in Table 3. The comparative analysis of these approaches is discussed in the conclusions section. The code used to generate PubMed embeddings and implement the RAG system is also publicly available[2,3].

3.1. Information Retrieval

3.1.1. Vector Database Construction

We downloaded the 2025 PubMed Annual Baseline [4]. We combined the titles and abstracts into a single text by adding a period and space (". ") after each title if not already present and prepending it to the abstract. If no title was present, only the abstract was used. If no abstract was present, only the title

[1] https://bioasq.org/
[2] https://github.com/ml-rmc/pubmed_embeddings
[3] https://github.com/ml-rmc/rag_qa/
[4] https://ftp.ncbi.nlm.nih.gov/pubmed/baseline/

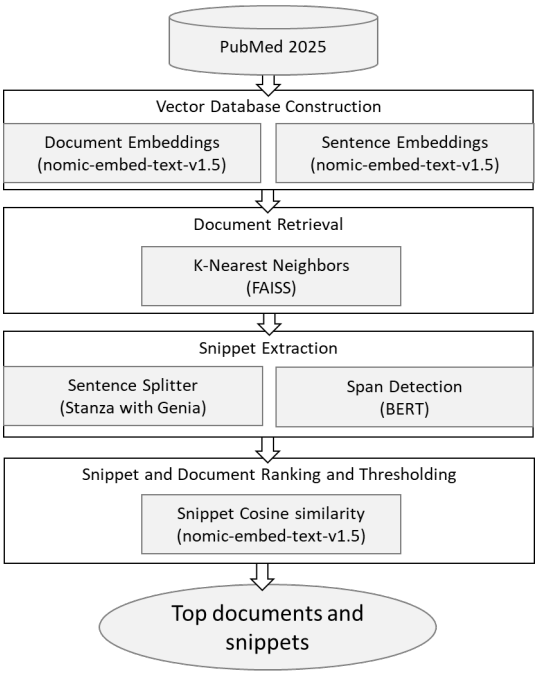

Figure 1: Overview of the information retrieval portion of our system.

was used. Titles and abstracts are cleaned by stripping trailing and leading whitespace, replacing new lines with spaces, and removing surrounding brackets ("[ ]") when present.

To generate embeddings, we used Nomic AI's `nomic-embed-text-v1.5` [4][5]. This model is designed to support long-context inputs of up to 8192 tokens and has demonstrated strong performance on both short-context benchmarks such as MTEB [17] and long-context benchmarks such as LoCo [18]. The model also produces Matryoshka embeddings [19], which support compressed vector representations. However, for BioASQ 2026 we used the full 768-dimensional embeddings.

We evaluated two approaches for context representation: document embeddings and sentence embeddings. After embedding generation, all vectors were normalized using `layernorm`, making Euclidean distance equivalent to cosine similarity during the subsequent K-nearest neighbors retrieval step. For document embeddings, the full document title and abstract were used as input to the embedding model. Although Nomic-Embed is designed for long-context inputs, prior work [20] suggests that shorter-context embeddings can often produce more accurate retrieval results even with models designed for long contexts. Motivated by this observation, we additionally experimented with sentence embeddings. To generate sentence embeddings, abstracts were first segmented using the biomedical Stanza [6][6] sentence splitter with the GENIA model. Each resulting sentence was then independently encoded using `nomic-embed-text-v1.5`.

This process produced two vector databases: one containing 38,197,689 document embeddings and another containing 209,479,852 sentence embeddings. These databases were used for PMID retrieval and, to support efficient retrieval of the associated abstracts, we organized abstract text into a hierarchical file system organized by PMID.

3.1.2. Document Retrieval

We retrieve relevant contexts using K-nearest neighbors (KNN) via the GPU implementation [21] of Facebook AI Similarity Search (Faiss) [5][7,8]. To do this, we first generate a question embedding using

[5]https://huggingface.co/nomic-ai/nomic-embed-text-v1.5
[6]https://stanfordnlp.github.io/stanza/biomed.html
[7]https://github.com/facebookresearch/faiss
[8]https://faiss.ai/index.html

nomic-embed-text-v1.5. Faiss computes the Euclidean distance between the question embedding and context embedding; however, because all embeddings are normalized using layernorm, Euclidean distance is equivalent to cosine similarity during retrieval.

We experimented with several values of K, but used K=100 for all BioASQ competition submissions. We treat document retrieval as a broad filtering step and rely on snippet extraction and reranking to filter this candidate document set downstream. The retrieval step therefore produces a ranked list of the 100 contexts most similar to the question embedding. Regardless of whether retrieval is performed using document or sentence embeddings, the final retrieved contexts are the title and abstract of a document. i.e. not the sentence that was embedded using sentence embeddings. Sentence embeddings are used solely to improve retrieval performance and are not directly returned as output contexts.

3.1.3. Snippet Extraction

We next perform snippet extraction from the 100 retrieved contexts. We evaluated three approaches for snippet retrieval: (1) sentence-based extraction, (2) BERT-based span detection, and (3) a hybrid approach combining both methods.

For sentence-based extraction, abstracts were segmented into sentences using Stanza sentence splitter with the GENIA model. All sentences are treated as candidate snippets and sent to the snippet ranking and thresholding step.

For the BERT-based span detection approach, we trained a bert-base-uncased model using the HuggingFace[9] AutoModelForQuestionAnswering framework. Training was performed using a learning rate of 2e-5, batch size of 90, weight decay of 0.01, and total of 5 epochs. Model selection was based on evaluation loss computed after each epoch, with the best-performing checkpoint restored at the end of training. We experimented with training for greater than 5 epochs, but found that overfitting generally started to occur after the second or third training epoch. We used the default tokenizer provided by AutoTokenizer.from_pretrained(bert-base-uncased). The model was trained using the BioASQ 14b dataset. To construct the training data, we retrieved the abstracts associated with each gold-standard snippet. During training, the question and abstract were concatenated and used as input to the model, which was trained to predict the start and end token positions of the gold-standard snippet. When the combined question and abstract exceeded BERT's maximum input length of 512 tokens, only the abstract portion was truncated. The span detection formulation imposes several limitations. First, only a single snippet can be predicted per abstract, which may not adequately capture all relevant evidence. Second, the 512-token input constraint may truncate long abstracts and potentially remove useful contextual information.

Lastly, for the hybrid approach, we combined the outputs of both sentence-based extraction and BERT-based span detection into a single candidate set prior to reranking.

3.1.4. Snippet and Document Ranking and Thresholding

For snippet filtering we first remove any empty strings. These were primarily generated as a result of the span-based extraction approach. Next, snippets were filtered based on semantic similarity to the question. To accomplish this, we generated embeddings for both the question and each candidate snippet using nomic-embed-text-v1.5 and computed the cosine similarity between them. Snippets with similarity scores below a threshold of 0.65 were discarded. The remaining snippets were then reranked according to their cosine similarity scores. This threshold was found via a manual tuning process where the ranked snippets of 20-30 questions were observed. Alternate strategies such as keeping the top 5 or 10 snippets were considered, but based on the same manual tuning process, we determined this would introduce too many uninformative snippets. Tuning of this threshold could be done in a more thorough systematic way in future iterations.

Because the filtering stage could still produce a large number of candidate snippets, we retained only the top K=10 ranked snippets, consistent with the BioASQ submission requirements. In cases where

[9] https://huggingface.co/

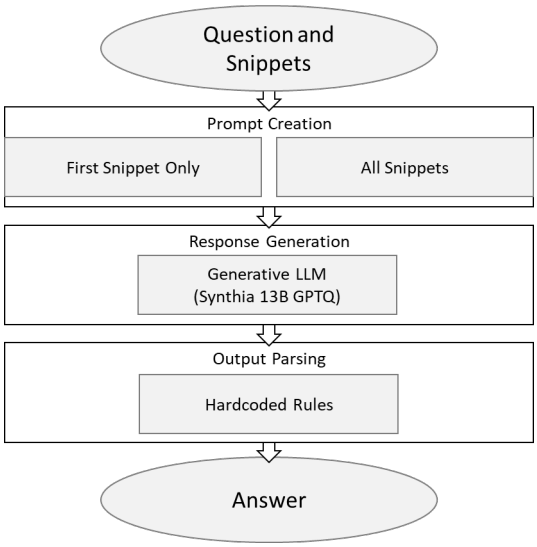

Figure 2: Overview of the answer generation portion of our system.

thresholding reduced the number of remaining snippets to fewer than 10, all remaining snippets were retained without additional filtering. Lastly, we edited our list of submitted documents by eliminating any documents for which no snippet remained. We ranked the documents based on the snippet rankings. In the case where multiple snippets were extracted from a single document, the document was ranked according to its highest ranked snippet.

3.2. Answer Generation

3.2.1. Prompt Creation

We use different prompt templates for different question types: `yesno`, `factoid`, `list`, `summary`, and `ideal`. The same prompt is used for both `summary` and `ideal` answers. Initial experimentation showed that placing the retrieved context before the question improved performance. This is consistent with prior work [22, 23, 24] showing that language models are often more influenced by information appearing later in the prompt, particularly when conflicting or competing contextual evidence is present [24]. Similarly, we found that repeating the desired output format near the end of the prompt and emphasizing important terms through formatting (e.g. all capital letters) improved performance [25]. The prompts for each question type are provided in Table 2.

3.2.2. Answer Generation

We used the Hugging Face Open LLM Leaderboards [10] to identify high-performing LLMs and evaluated several candidates on past BioASQ datasets. Due to computational constraints, we limited our search to models containing at most 13 billion parameters. Based on preliminary experimentation, we selected `Synthia-13B-GPTQ` [8][11] as our primary model.

`Synthia-13B-GPTQ` is based on the LLaMA 2 architecture [9] and was further trained on Orca-style instruction-following datasets [26]. The model is designed for instruction following and long-form conversational generation. We used the Gradient Post-Training Quantization (GPTQ) 4-bit version of the model.

[10] https://huggingface.co/open-llm-leaderboard
[11] https://huggingface.co/Synthia-13B-GPTQ

Prompt Type	Prompt Text

YesNo	`You are a biomedical question answering assistant.`
	`A user will ask a question, and you should provide a YES OR NO answer using the provided context.`
	`ONLY generate an answer from the provided context.`
	`Context: <context>`
	`Question: <question>`
	`ONLY answer with a YES OR NO`
	`Answer:`

Factoid	`You are a biomedical question answering assistant.`
	`A user will ask a question, and you should provide a ONE TERM answer using the provided context.`
	`ONLY generate an answer from the provided context.`
	`Context: <context>`
	`ONLY return a one term answer and nothing more.`
	`Question: <question>`
	`Answer:`

List	`You are a biomedical question answering assistant.`
	`A user will ask a question, and you should provide a LIST of FIVE terms using the provided context.`
	`ONLY generate an answer from the provided context.`
	`Context: <context>`
	`ONLY return a comma seperated LIST of FIVE terms`
	`Question: <question>`
	`Answer:`

Summary / Ideal	`You are a biomedical question answering assistant.`
	`A user will ask a question, and you should provide a concise answer based on the following context.`
	`Context: <context>`
	`Question: <question>`
	`Answer:`

Table 2
Prompts for each question type

3.2.3. Output Parsing

Output parsing is a critical step but also highly dependent on the specific LLM. As a result, this stage requires iterative manual inspection and refinement. Across all question types, we apply several standard post-processing steps: (1) removing the original prompt text so that only the generated response is processed, (2) removing padding tokens (</s> and <pad>), (3) removing new line characters, and (4) removing unknown tokens (<unk>). We perform no additional output parsing for `factoid`, `summary`, and `ideal` questions. Extra steps for `yesno` and `list` questions are provided below.

For `yesno` questions, we convert the response to a *yes* or *no* by checking if the word *yes* or *no* exists anywhere in the answer and output a *yes* or *no* respectively. When both a *yes* and *no* exist in the answer or neither a *yes* nor a *no* exist in the answer, we output a *yes*. This heuristic is based on our observation that the majority of answers are *yes* responses (1123 *yes* versus 418 *no* in BioASQ 14b training data).

For `list` questions, we extract items from various forms of generated lists. We first convert the text to lowercase, replace new lines with a comma, and replace `digit period space` patterns with a comma (e.g. "1. value1 2. value2" => ",value1, value2"). This can result in two commas, so we replace double commas with a single comma. Next we split the text on commas which results in an inconsistent list of items which we process one at a time. For each item, we replace various remnants of list formatting such as "1)", "1", "and", "or" with empty strings. Lastly we remove trailing punctuation, trailing or leading white space, and items which are empty strings.

3.3. Evaluation Criteria

We used the official evaluation metrics as described by Balikas et al [27]. We summarize them in Tables 3, 4, and 5. All metrics are averaged over all questions. `Summary` and `ideal` questions are combined in the same evaluation and use both automatic and manual scores. Automatic scores are based on ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [28]. ROUGE is a set of metrics used for evaluating automatic summarization. The metrics compare an automatically produced summary against a gold standard reference summary. ROUGE-2 considers overlap of bigrams (n=2). ROUGE-SU4 considers overlap of skip bigrams which allows up to a distance of 4 in-between words. i.e. the bigrams don't have to be continuous and may be separated by up to 4 words. Manual evaluation scores were not available at time of publication.

Abbreviation	Description
Mean Prec	Mean Precision: average proportion of retrieved items that are relevant
Mean Recall	Mean Recall: average proportion of relevant items that were successfully retrieved
F-Measure	Mean F-Measure: average harmonic mean of precision and recall
MAP*	Mean Average Precision: mean of the average precision scores across all queries
GMAP	Geometric Mean Average Precision: geometric mean of average precision scores across all queries

Table 3
Definitions of information retrieval evaluation metrics. * indicates primary evaluation metric

Question Type	Abbreviation	Description
Yes/No	Acc	Accuracy
	F1Yes	F1 score for *yes* questions
	F1No	F1 score for *no* questions
	MaF1*	Macro F1 score between *yes* and *no* questions.
Factoid	StrAcc*	Strict accuracy: prediction is counted correct only if predicted answer exactly matches the gold answer
	LenAcc	Lenient accuracy: prediction is counted correct if the predicted answer overlaps with the gold answer
	MRR*	Mean Reciprocal Rank: average reciprocal rank of the first correct answer in the returned list of predictions. Where the reciprocal rank is defined as $1/$ (rank of first correct answer)
List	Prec	Mean Precision: average proportion of retrieved answers that are relevant
	Rec	Mean Recall: average proportion of relevant answers that were successfully retrieved
	F1**	Mean F1: Harmonic mean of precision and recall

Table 4
Definitions of answer generation metrics. * indicates the primary evaluation metric for that question type.

4. Results

In this section we present our results for Task 14b Phases, A, A+, and B. All results are the officially reported preliminary results (available online[12]). Official final results will be available after the official conference which is after the publication of this paper. Results are presented in Tables 6, 7, 8, 9, 10, and 11. Bold terms indicate the highest primary evaluation metric for that batch. Results for Synergy

[12] https://participants-area.bioasq.org/

Abbreviation	Description
Automatic scores	
R-2 (Rec)	Rouge-2 Recall: the percentage of gold standard bigrams in the generated answer
R-2 (F1)	Rouge-2 F1: balances precision and recall of bigrams in the generated answer
R-SU4 (Rec)	Rouge-SU4 Recall: the percentage of gold standard bigrams in the generated answer but allows for skips up to a distance of 4
R-SU4 (F1)	Rouge-SU4 F1: balances precision and recall of skip bigrams in the generated answer
Manual scores*	
Readability	The answer is easily readable and fluent
Recall	All the necessary information is reported
Precision	No irrelevant information is reported
Repetition	The answer does not repeat the same information multiple times

Table 5
Definitions of ideal and summary question evaluation. *All manual scores are used as the official evaluation metric. Descriptions for manual scores stated verbatim from [27].

are not reported, but are available online. Our system performed poorly on all rounds of Synergy except for round 1 for which its performance was moderate.

	Run	Mean Precision	Recall	F-Measure	MAP*	GMAP
1	RMC_1	0.0484	0.1704	0.0690	0.1042	0.0003
	RMC_2	0.1035	0.1270	0.1016	**0.0998**	0.0001
2	RMC_1	0.0276	0.1367	0.0429	0.0881	0.0001
	RMC_2	0.0813	0.1141	0.0873	0.0995	0.0001
	RMC_3	0.0434	0.1570	0.0604	0.1069	0.0002
	RMC_4	0.0263	0.1191	0.0405	0.0856	0.0001
	RMC_5	0.0485	0.1456	0.0640	**0.1090**	0.0001
3	RMC_1	0.0320	0.0919	0.0454	0.0374	0.0001
	RMC_2	0.0583	0.0631	0.0561	0.0511	0.0000
	RMC_3	0.0320	0.0919	0.0454	0.0374	0.0001
	RMC_4	0.0388	0.1640	0.0595	0.1099	0.0002
	RMC_5	0.0406	0.1807	0.0629	**0.1224**	0.0002
4	RMC_1	0.0441	0.1085	0.0572	0.0691	0.0001
	RMC_2	0.1028	0.1006	0.0896	0.0688	0.0001
	RMC_3	0.0497	0.1261	0.0617	**0.0707**	0.0002
	RMC_4	0.0200	0.0607	0.0293	0.0339	0.0000
	RMC_5	0.0394	0.0885	0.0516	0.0533	0.0001

Table 6
Task 14b Phase A Document retrieval results.

For Phase A Documents we achieved competitive results, but were not a top performer. Our main problem is over-generation of results. While many of our retrieved documents are correct, we also retrieved many incorrect documents. These documents tend to be relevant to the general topic, but lack specific details required to answer the question. These incorrect documents should get filtered in subsequent stages and indicate that future work in improving snippet and document filtering is required.

Successful snippet extraction required successful document retrieval, so not surprisingly, our snippet extraction performance was generally low. RMC_2 did, however, perform the best among all systems and teams for Phase A snippets batch 2. Manual analysis indicates that our system is capable of extracting high quality snippets, but these snippets are buried within many lower quality snippets. Again, this indicates that future work in snippet ranking and filtering is necessary.

	Run	Mean Prec	Recall	F-Measure	MAP*	GMAP
1	RMC_1	0.0233	0.0764	0.0254	0.0357	0.0001
	RMC_2	0.0360	0.1036	0.0436	**0.1113**	0.0001
2	RMC_1	0.0115	0.0488	0.0156	0.0377	0.0000
	RMC_2	0.0373	0.0846	0.0419	**0.2108**	0.0001
	RMC_3	0.0192	0.0613	0.0247	0.0650	0.0001
	RMC_4	0.0188	0.0509	0.0204	0.0390	0.0000
	RMC_5	0.0315	0.0760	0.0361	0.0763	0.0001
3	RMC_1	0.0110	0.0163	0.0107	0.0064	0.0000
	RMC_2	0.0190	0.0369	0.0200	**0.0539**	0.0000
	RMC_3	0.0110	0.0163	0.0107	0.0064	0.0000
	RMC_4	0.0149	0.0444	0.0187	0.0345	0.0001
	RMC_5	0.0183	0.0516	0.0234	0.0366	0.0001
4	RMC_1	0.0302	0.0196	0.0125	0.0190	0.0000
	RMC_2	0.0380	0.0572	0.0352	**0.0528**	0.0001
	RMC_3	0.0244	0.0266	0.0147	0.0250	0.0001
	RMC_4	0.0025	0.0028	0.0017	0.0031	0.0001
	RMC_5	0.0032	0.0092	0.0029	0.0056	0.0000

Table 7
Task 14b Phase A Snippets results. Bold terms indicate the highest score for that batch. Since all MAP scores are 0, we use F-measure as our primary evaluation metric, however MAP is the official BioASQ primary metric.

Manual analysis indicates that our system will highly rank snippets which match the broad concepts (are semantically similar), but lack specific details required to answer the specific question. For example, for the question “What is the recurrence percentage of Cushings Disease?”, the top ranked snippet is "There is no consensus on the remission criteria for Cushing’s disease or on the definition of disease recurrence after transsphenoidal surgery, and comparison of the different published series is therefore difficult”. This is relevant, but contains no percentage which can be extracted for the answer. Similarly, the second ranked item is “A long-term recurrence rate of Cushing’s disease ranging from 2%-25% has been reported.”, which contains too broad a range to answer the question. Snippets after these top two are less relevant.

Despite the relatively poor retrieval performance of our system, the results for Phase A+ were competitive, often ranking in the middle or upper tier of submissions. This observation aligns with prior work showing that LLMs can generate strong answers even when provided with limited or uninformative retrieved context [3, 29]. We did not make any submission for RMC_3 for the first batch due to time constraints.

For Task B we see an expected increase in performance from Phase A+. This aligns with research indicating that good context increases performance [30]. Our performance generally fell within the middle or upper-tier of rankings. Based on an error analysis, two problems immediately stood out: (1) Problems with questions about numbers. Our model would often hallucinate a number despite having the correct number in the context. (2) Formatting issues with lists which could be corrected through different prompting or output parsing.

5. Conclusions

In this paper, we described our submissions to the 2026 BioASQ challenge, Tasks Synergy and 14b. Our approach was a local RAG system that combined dense vector retrieval with a LLaMA 2-based LLM. Our system achieved moderate performance. The system consisted of three primary components: document retrieval, snippet extraction, and answer generation. Across these components, we evaluated several alternative approaches, and our findings are summarized below.

Embedding Types: Document versus Sentence: Sentence embeddings (RMC_2 and RMC_3) performed the best in half of the batches (batches 1 and 4), while document embeddings (RMC_5)

		Yes/No				Factoid			List		
	Run	Acc	F1Yes	F1No	MaF1*	StrAcc	LenAcc	MRR*	Prec	Rec	F1*
1	RMC_1	0.7647	0.7778	0.7500	**0.7639**	0.0000	0.0000	0.0000	0.0714	0.0532	0.0540
	RMC_2	0.5882	0.4615	0.6667	0.5641	0.1739	0.1739	0.1739	0.1365	0.0831	0.1001
	RMC_3	-	-	-	-	-	-	-	-	-	-
	RMC_4	0.6471	0.6667	0.6250	0.6458	0.0870	0.0870	0.0870	0.1333	0.0802	0.0972
	RMC_5	0.7647	0.7778	0.7500	**0.7639**	0.2174	0.2174	**0.2174**	0.1222	0.1009	**0.1021**
2	RMC_1	0.7619	0.8148	0.6667	0.7407	0.1000	0.1000	0.1000	0.1385	0.1454	0.1305
	RMC_2	0.8095	0.8462	0.7500	0.7981	0.1000	0.1000	0.1000	0.1436	0.1064	0.1170
	RMC_3	0.7619	0.8148	0.6667	0.7407	0.1000	0.1000	0.1000	0.1385	0.1454	0.1305
	RMC_4	0.7619	0.8000	0.7059	0.7529	0.0000	0.0000	0.0000	0.1538	0.1699	0.1486
	RMC_5	0.8571	0.8889	0.8000	**0.8444**	0.1000	0.1000	0.1000	**0.3051**	0.2490	**0.2610**
3	RMC_1	0.8182	0.8571	0.7500	0.8036	0.0588	0.0588	0.0588	0.2353	0.2239	**0.2179**
	RMC_2	0.6364	0.6667	0.6000	0.6333	0.1765	0.1765	0.1765	0.1471	0.1176	0.1210
	RMC_3	0.8182	0.8571	0.7500	0.8036	0.0588	0.0588	0.0588	0.2353	0.2239	**0.2179**
	RMC_4	0.9091	0.9333	0.8571	**0.8952**	0.2353	0.2353	0.2353	0.1690	0.1701	0.163
	RMC_5	0.9091	0.9333	0.8571	**0.8952**	0.3529	0.3529	**0.3529**	0.1500	0.1520	0.1422
4	RMC_1	0.6875	0.7619	0.5455	0.6537	0.2727	0.2727	**0.2727**	0.2875	0.2493	0.2620
	RMC_2	0.7500	0.8000	0.6667	**0.7333**	0.1818	0.1818	0.1818	0.1443	0.1339	0.1337
	RMC_3	0.7500	0.8182	0.6000	0.7091	0.2727	0.2727	**0.2727**	0.2375	0.2027	0.2138
	RMC_4	0.6875	0.7826	0.4444	0.6135	0.2727	0.2727	**0.2727**	0.2700	0.2610	**0.2506**
	RMC_5	0.7500	0.8182	0.6000	0.7091	0.0909	0.0909	0.0909	0.1967	0.1780	0.1723

Table 8
Task 14b Phase A+ results for Batches 1-4. Bold terms indicate the highest score for that batch.

	Run	R-2 (Rec)	R-2 (F1)	R-SU4 (Rec)	R-SU4 (F1)
1	RMC_1	0.1327	0.1235	0.1494	0.1331
	RMC_2	0.1384	**0.1429**	0.1541	**0.1497**
	RMC_3	-	-	-	-
	RMC_4	0.1538	0.1294	0.1623	0.1331
	RMC_5	0.1396	0.1259	0.1562	0.1365
2	RMC_1	0.1807	0.1704	0.1852	0.1735
	RMC_2	0.1730	**0.1822**	0.1768	**0.1851**
	RMC_3	0.1814	0.1708	0.1860	0.1741
	RMC_4	0.2020	0.1681	0.2017	0.1658
	RMC_5	0.2001	0.1619	0.2112	0.1628
3	RMC_1	0.1307	0.1317	0.1270	0.1288
	RMC_2	0.1293	**0.1434**	0.1217	**0.1336**
	RMC_3	0.1307	0.1310	0.1270	0.1281
	RMC_4	0.1349	0.1247	0.1359	0.1236
	RMC_5	0.1429	0.1338	0.1459	0.1310
4	RMC_1	0.1606	0.1574	0.1536	0.1507
	RMC_2	0.1522	**0.1643**	0.1484	**0.1616**
	RMC_3	0.1595	0.1577	0.1562	0.1536
	RMC_4	0.1651	0.1384	0.1668	0.1386
	RMC_5	0.1760	0.1527	0.1790	0.1555

Table 9
Task 14b Phase A+ ROUGE Results. Evaluation for `summary` and `ideal` questions are combined. Bold terms indicate the highest score for that batch. Manual results were not available at time of publication.

performed the best in the other half (batches 2 and 3). This makes drawing conclusions difficult and we will continue to explore both methods in future systems.

Snippet Extraction: Sentence Splitting versus Span Detection: RMC_2 performed the best for all snippet retrieval batches and was the top among all systems and teams (based on MAP) for round 2. RMC_2 relied on sentence splitting, indicating a simple sentence splitting strategy is preferred over

	Run	Yes/No				Factoid			List		
		Acc	F1Yes	F1No	MaF1*	StrAcc	LenAcc	MRR*	Prec	Rec	F1*
1	RMC_1	0.8235	0.8421	0.8000	0.8211	0.2174	0.2174	0.2174	0.2032	0.1491	0.1618
	RMC_2	0.8824	0.9000	0.8571	**0.8786**	0.4348	0.4348	**0.4348**	0.2613	0.2379	**0.2342**
2	RMC_1	0.8571	0.8889	0.8000	0.8444	0.0500	0.0500	0.0500	0.2728	0.2858	0.2669
	RMC_2	0.9048	0.9231	0.8750	**0.8990**	0.2500	0.2500	**0.2500**	0.3692	0.3516	**0.3438**
3	RMC_1	0.9091	0.9333	0.8571	**0.8952**	0.1765	0.1765	0.1765	0.3765	0.2614	0.2970
	RMC_2	0.9091	0.9333	0.8571	**0.8952**	0.4706	0.4706	**0.4706**	0.3871	0.3498	**0.3530**
4	RMC_1	0.7500	0.8000	0.6667	0.7333	0.3636	0.3636	0.3636	0.3600	0.2834	0.3032
	RMC_2	0.8750	0.8889	0.8571	**0.8730**	0.5455	0.5455	**0.5455**	0.5500	0.4817	**0.4943**

Table 10
Task 14b Phase B results. Bold terms indicate the highest score for that batch.

	Run	R-2 (Rec)	R-2 (F1)	R-SU4 (Rec)	R-SU4 (F1)
1	RMC_1	0.1880	0.1910	0.1937	0.1888
	RMC_2	0.2218	**0.2141**	0.2215	**0.2063**
2	RMC_1	0.2311	**0.2305**	0.2194	**0.2161**
	RMC_2	0.2684	0.2303	0.2529	0.2149
3	RMC_1	0.1862	0.2024	0.1738	0.1896
	RMC_2	0.2367	**0.2278**	0.2224	**0.2144**
4	RMC_1	0.2050	0.2064	0.1953	0.1927
	RMC_2	0.2779	**0.2561**	0.2703	**0.2474**

Table 11
Task 14b Phase B ROUGE Results. Evaluation for `summary` and `ideal` questions are combined. Bold terms indicate the highest score for that batch. Manual results were not available at time of publication.

more complex span detection models. This simplifies our pipeline, but places an increased importance on the snippet ranking and thresholding step.

Prompt Context: First Only versus All Snippets: RMC_2, which uses all snippets rather than just the first performed the best for all rounds and all question types of Phase B. More mixed results were observed for Phase A+. This is likely due to conflicting or lower-quality snippets being present in Phase A+. Therefore implying that using all snippets is preferable with consistent high quality snippets, but can decrease performance with lower-quality and conflicting contexts. This raises an interesting question about real-world deployable systems where the quality of retrieved contexts cannot be guaranteed.

6. Future Work

These experiments represent an initial step in the development of our system and highlights several directions for improvement. In future work, we plan to:

- Optimize document retrieval hyperparameters and explore more embeddings types.
- Critically we must improve snippet ranking and thresholding. This may require an exploration of different embedding types or incorporating question type and answer-type awareness.
- Investigate context ordering strategies. Our current implementation presents snippets in descending retrieval order, but alternative orderings may improve answer quality.
- Perform a broader search of large language models, including biomedical-specific models.
- Refine prompt creation and output parsing, particularly with respect to `list` questions.
- Reduce hallucinations for numeric questions, potentially through extractive approaches.

7. Acknowledgments

This work was funded in part by a Chenery Research Professorship grant awarded by Randolph-Macon College.

8. Declaration on Generative AI

We used ChatGPT to summarize notes in the early stages of drafting the paper and to help format tables. All text has been reviewed and edited for correctness, clarity, and conciseness. The authors take full responsibility for the content.

References

[1] A. Nentidis, G. Katsimpras, A. Krithara, M. Krallinger, M. Rodríguez-Ortega, E. Rodriguez-López, N. Loukachevitch, I. Rozhkov, E. Tutubalina, D. Dimitriadis, V. Patsiou, G. Tsoumakas, G. Giannakoulas, A. Bekiaridou, A. Samaras, G. M. Di Nunzio, N. Ferro, S. Marchesin, M. Martinelli, G. Silvello, G. Paliouras, Overview of BioASQ 2026: The fourteenth BioASQ Challenge on Large-Scale Biomedical Semantic Indexing and Question Answering, volume Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Seventeenth International Conference of the CLEF Association (CLEF 2026) of *Lecture Notes in Computer Science*, Springer, 2026.

[2] A. Nentidis, G. Katsimpras, A. Krithara, G. Paliouras, Overview of BioASQ Tasks 14b and Synergy14 in CLEF2026, in: E. Sánchez Salido, A. Barrón-Cedeño, A. García Seco de Herrera, S. MacAvaney, J. M. Struß (Eds.), CLEF 2025 Working Notes, 2026.

[3] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, Advances in Neural Information Processing Systems 33 (2020) 9459–9474.

[4] Z. Nussbaum, J. X. Morris, B. Duderstadt, A. Mulyar, Nomic embed: Training a reproducible long context text embedder, arXiv preprint arXiv:2402.01613 (2024).

[5] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, H. Jégou, The faiss library (2024). arXiv:2401.08281.

[6] Y. Zhang, Y. Zhang, P. Qi, C. D. Manning, C. P. Langlotz, Biomedical and clinical english model packages for the stanza python nlp library, Journal of the American Medical Informatics Association 28 (2021) 1892–1899.

[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of naacl-hlt, 2019, pp. 4171–4186. URL: https://arxiv.org/abs/1810.04805.

[8] M. Tissera, Synthia-13b: Synthetic intelligent agent, https://huggingface.co/migtissera/Synthia-13B, 2023.

[9] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., Llama 2: Open foundation and fine-tuned chat models, arXiv preprint arXiv:2307.09288 (2023).

[10] P. Rajpurkar, J. Zhang, K. Lopyrev, P. Liang, SQuAD: 100,000+ questions for machine comprehension of text, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 2383–2392. URL: https://aclanthology.org/D16-1264. doi:10.18653/v1/D16-1264.

[11] S. Fox, M. Duggan, Health online 2013, Pew Internet and American Life Project 2013 (2013) 1–55.

[12] D. L. Sackett, Evidence-based medicine, in: Seminars in perinatology, volume 21, 1997, pp. 3–5.

[13] T. Russell-Rose, Chamberlain, Expert search strategies: the information retrieval practices of healthcare information professionals, JMIR medical informatics 5 (2017) e33.

[14] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos, et al., An overview of the bioasq large-scale biomedical semantic indexing and question answering competition, BMC bioinformatics 16 (2015) 1–28.

[15] D. L. Wheeler, T. Barrett, D. A. Benson, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. DiCuccio, R. Edgar, S. Federhen, et al., Database resources of the national center for biotechnology information, Nucleic acids research 35 (2007) D5–D12.

[16] A. Krithara, A. Nentidis, K. Bougiatiotis, G. Paliouras, Bioasq-qa: A manually curated corpus for biomedical question answering, Scientific Data 10 (2023) 170.

[17] N. Muennighoff, N. Tazi, L. Magne, N. Reimers, Mteb: Massive text embedding benchmark, in: Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, 2023, pp. 2014–2037.

[18] J. Saad-Falcon, D. Y. Fu, S. Arora, N. Guha, C. Ré, Benchmarking and building long-context retrieval models with loco and m2-bert, arXiv preprint arXiv:2402.07440 (2024).

[19] A. Kusupati, G. Bhatt, A. Rege, M. Wallingford, A. Sinha, V. Ramanujan, W. Howard-Snyder, K. Chen, S. Kakade, P. Jain, et al., Matryoshka representation learning, Advances in Neural Information Processing Systems 35 (2024) 30233–30249.

[20] M. Günther, I. Mohr, D. J. Williams, B. Wang, H. Xiao, Late chunking: contextual chunk embeddings using long-context embedding models, arXiv preprint arXiv:2409.04701 (2024).

[21] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, IEEE Transactions on Big Data 7 (2019) 535–547.

[22] A. Peysakhovich, A. Lerer, Attention sorting combats recency bias in long context language models, arXiv preprint arXiv:2310.01427 (2023).

[23] B. Veseli, J. Chibane, M. Toneva, A. Koller, Positional biases shift as inputs approach context window limits, arXiv preprint arXiv:2508.07479 (2025).

[24] B. Zhang, A. Bornet, R. Yang, N. Liu, D. Teodoro, Healthcontradict: Evaluating biomedical knowledge conflicts in language models, npj Digital Medicine (2026).

[25] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, G. Neubig, Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing, ACM computing surveys 55 (2023) 1–35.

[26] S. Mukherjee, A. Mitra, G. Jawahar, S. Agarwal, H. Palangi, A. Awadallah, Orca: Progressive learning from complex explanation traces of gpt-4, 2023. arXiv:2306.02707.

[27] G. Balikas, I. Partalas, A. Kosmopoulos, S. Petridis, P. Malakasiotis, I. Pavlopoulos, I. Androutsopoulos, N. Baskiotis, E. Gaussier, T. Artieres, et al., Evaluation framework specifications, Project deliverable D 4 (2013).

[28] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, in: Text summarization branches out, 2004, pp. 74–81.

[29] J. Li, J. Wang, Z. Zhang, H. Zhao, Self-prompting large language models for zero-shot open-domain qa, arXiv preprint arXiv:2212.08635 (2022).

[30] F. Shi, X. Chen, K. Misra, N. Scales, D. Dohan, E. H. Chi, N. Schärli, D. Zhou, Large language models can be easily distracted by irrelevant context, in: International Conference on Machine Learning, PMLR, 2023, pp. 31210–31227.