Biomedical Semantic Question Answering - Answering Systems Using Different LLMs for Subtasks

Notebook for the BioASQ Lab at CLEF 2025

Samitinjaya[1,*,†], Dipankar Das[2,†]

Jadavpur University, Kolkata, India

Abstract

This work presents our systems designed for the BioASQ Task13b, in which we tried to create base systems using sentence tranformers and Large Language models(LLMs) to address the task of BioASQ Task13b. The systems aims to address diverse biomedical questions by integrating Information Retrieval (IR), extractive QA, and summarization. Information retrieval part covers extracting related articles(documents) and then return them with a decreasing order of relevance. The article retrieval is done by the help of Bio.Entrez module. The article ranking for relevance is done using the sentence-transformers/all-MiniLM-L6-v2 model. Snippets are to be retrieved from each document that contain the related answer or part to the question. For snippet extraction deepset/roberta-base-squad2, has been used. Exact and Ideal answers extractive and generative answer respectively for the question. Both of these has been done using a fine-tuned T5 model (google-t5/t5-base) .

The problem statement has been explained for an easy understanding of each of the phase in the challenge along with the required answers in each of these phases. Then a brief overview of the approach used has been provided. For easy understanding of the challenge, the 3 phases of the challenge can be divided by us into multiple subtasks, each one solving a particular part of the challenge. An overview of how the systems are created and how subtasks are approached has been provided. Implementation part describes how each of these subtask has been implemented to create systems that can easily solve the challenge phases. Framework of the system is given for an in-depth understanding.

The official preliminary results from the BioASQ Task13b for the Test data 4 achieved by our systems has been shared here. The organizers are yet to perform a manual evaluation. The results were based on an automated evaluation.

The overall performance of our systems remain limited, however the systems are functional, and since each part in system can be treated separately, so we can easily improve the parts individually to get a better output. Our system lays a strong foundation for future enhancements. Future work can focus on improving document relevance, integrating domain-specific knowledge, and optimizing answer generation quality.

Keywords

BioASQ Task 13b, System Approach, System Framework, CEUR-WS

CEUR-WS.org/Vol-4038/paper_41.pdf

1. Introduction

The rapid growth of the biomedical literature presents both opportunities and challenges in the quest to access relevant and accurate information efficiently. With millions of new citations being added annually to databases like PubMed[1], practitioners and researchers often face difficulties in retrieving precise answers and comprehensive summaries from this vast and unstructured data. Traditional keyword-based search methods are limited in addressing issues such as synonymy and contextual relevance, which are prevalent in biomedical terminology.

In this paper we have shared our approach to solve the issues stated above.

Semantic Question Answering (QA) systems have emerged as a vital solution to these challenges by leveraging advanced natural language processing (NLP) techniques or machine learning to understand, retrieve and generate relevant answers. These systems can go beyond simple keyword matching by

CLEF 2025 Working Notes, 9 – 12 September 2025, Madrid, Spain

[*]Corresponding author.

[†]These authors contributed equally.

samitgupta03@gmail.com (Samitinjaya); dipankar.dipnil2005@gmail.com (D. Das)

© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1]https://pubmed.ncbi.nlm.nih.gov/

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

interpreting the underlying intent of questions and provide contextually appropriate answers to the user.

Within this scope, the BioASQ[2] challenge has been instrumental in promoting research and development in biomedical semantic indexing, information retrieval, question answering, and summarization. BioASQ Task 13b[3] focuses on generating accurate and comprehensive answers to biomedical questions, requiring integration of retrieval and summarization techniques to provide the answers

This work aims to develop and implement methods that effectively integrate information retrieval, answer extraction, and summarization processes to tackle complex biomedical queries. By participating in and addressing the challenges posed by BioASQ Task 13b, the work contributes to advancing the capabilities of automated systems in providing reliable, precise, and concise biomedical knowledge, ultimately supporting researchers, and healthcare professionals in decision-making processes.

2. Background And Motivation

The exponential growth of biomedical literature, with millions of articles in repositories such as PubMed, poses significant challenges for researchers and clinicians seeking quick and precise information. As the volume of available knowledge continues to expand rapidly, it becomes increasingly difficult to efficiently retrieve relevant, evidence-based answers to complex biomedical questions using traditional search methods.

Conventional keyword-based search systems often fall short in capturing the true semantic intent behind user queries, leading to retrieval of irrelevant information or missing critical data. This gap underscores the need for advanced systems capable of understanding, reasoning, and summarizing biomedical knowledge effectively.

Aim here is to develop systems that can bridge this gap by integrating information retrieval, natural language understanding, and summarization techniques to support the biomedical community more effectively. The task aims to tackle multiple problems which address the critical need for automated and scalable solutions to manage the overwhelming influx of biomedical data.

3. Outline Of The Paper

This paper gives us an overview on how the task has been approached and done. This paper has been organized as follows.

In the Introduction part, the details and overview of the challenge has been provided. It briefly describes what this challenge is about and what are the aims this challenge wants to achieve. Then background and motivation for this work is discussed.

Next, all the details related to the task are described in detail. The problem statement describes the problem/challenge/task. Then a detailed description of task is provided. Then it deep dives into what approach has been used by us to solve this problem. Each part of the approach is elaborated to give a detailed overview of the techniques used.

The section Implementation describes about how the methodology has been implemented. The structure of our systems after implementing every part is given here. The Results section gives an insight on the performance and overall usefulness of the implementation. The discussion section then provides us with the interpretations of the Results along with various improvements that can be made in system.

At last the paper has a Conclusions and Future Work section where the conclusions of the project are given along with some ideas of improvement and future related work that can be done. After that the references used are provided.

[2]https://www.bioasq.org/
[3]https://participants-area.bioasq.org/general_information/Task13b/

4. Problem Statement

To develop a system/s that can accurately retrieve relevant biomedical documents and generate precise and comprehensive answers to biomedical questions of different types, thereby reducing the manual effort, inefficiency, and limitations of traditional literature search methods.

5. Detailed Description Of Task

To address this challenge, BioASQ Task 13b provides a benchmark task that promotes the development of Biomedical Semantic Question Answering (QA) systems capable of automatically handling such queries. The task focuses on four types of questions : factoid, list, yes/no, and summary.

The system under development is required to:

- Retrieve relevant documents from biomedical repositories (e.g., PubMed)
- Extract relevant snippets of text from the retrieved documents
- Extract exact answers (facts, lists, yes/no)
- Generate ideal answers (concise, paragraph-sized summaries)

The BioASQ Task 13b organizes this into three phases, each representing a key sub-task:

- Phase A: The system retrieves up to 10 relevant documents, ranked by decreasing relevance, and returns snippets—text spans that may help answer the question (typically extracted from abstracts).
- Phase A+: No gold-standard documents/snippets are provided. Participants must use their Phase A outputs to directly extract Exact Answers and generate Ideal Answers.
- Phase B: Gold-standard relevant documents and snippets (manually curated) are provided to the participants. Using these high-quality resources, the task is to extract Exact Answers and generate Ideal Answers.

5.1. Required answers in each of the Phase :

5.1.1. In Phase A of Task 13b

In Phase A of Task 13b, the participants will be provided with English questions q_1, q_2,...,q_n. For each question q_i, each participating system will be required to return any (ideally all) of the following lists:

- A list of at most 10 relevant articles (documents) $d_{i,1}$, $d_{i,2}$, $d_{i,3}$,... from the designated article repositories. Again, the list should be ordered by decreasing confidence, i.e., $d_{i,1}$, should be the article that the system considers to be the most relevant to the question q_1,, $d_{i,2}$, should be the article that the system considers to be the second most relevant etc. A single article list will be returned per question and participating system, and the list may contain articles from multiple designated repositories. The returned article list will actually contain unique article identifiers (obtained from the repositories).
- A list of at most 10 relevant text snippets $s_{i,1}$, $s_{i,2}$, $s_{i,3}$,... from the returned articles. Again, the list should be ordered by decreasing confidence. A single snippet list will be returned per question and participating system, and the list may contain any number (or no) snippets from any of the returned articles $d_{i,1}$, $d_{i,2}$, $d_{i,3}$,... Each snippet will be represented by the unique identifier of the article it comes from, the identifier of the section the snippet starts in, the offset of the first character of the snippet in the section the snippet starts in, the identifier of the section the snippet ends in, and the offset of the last character of the snippet in the section the snippet ends in. The snippets themselves will also have to be returned (as strings)

5.1.2. In Phase A+ of Task 13b

In Phase A+ of Task 13b, the participants will be provided with English questions as in Phase A (above), but will be required to return "exact" and/or "ideal" answers, as in for Phase B (below).

5.1.3. In Phase B of Task 13b

In Phase B of Task 13b, the participants will be provided with the same questions q_1, q_2,...,q_n as in Phase A, but this time they will also be given gold (correct) lists of articles and snippets. The "gold" lists will contain articles and snippets identified by biomedical experts as relevant and providing enough information to answer the questions. For each question, each participating system may return an ideal answer, i.e., a paragraph-sized summary of relevant information. In the case of yes/no, factoid, and list questions, the systems may also return exact answers; for summary questions, no exact answers will be returned. The participants will be told the type of each question. A participating system may return only "exact" answers, or only "ideal" answers, or (ideally) both "exact" and "ideal" answers.

Exact Answers

- For each yes/no question, the exact answer of each participating system will have to be either "yes" or "no".
- For each factoid question, each participating system will have to return a list* of up to 5 entity names (e.g., up to 5 names of drugs), numbers, or similar short expressions, ordered by decreasing confidence.
- For each list question, each participating system will have to return a single list* of entity names, numbers, or similar short expressions, jointly taken to constitute a single answer (e.g., the most common symptoms of a disease). The returned list will have to contain no more than 100 entries of no more than 100 characters each.
- No exact answers will be returned for summary questions.

Ideal Answers

For each question (yes/no, factoid, list, summary), each participating system of Phase B may also return an ideal answer, i.e., a single paragraph-sized text ideally summarizing the most relevant information from articles and snippets retrieved in Phase A. Each returned "ideal" answer is intended to approximate a short text that a biomedical expert would write to answer the corresponding question (e.g., including prominent supportive information), whereas the "exact" answers are only "yes"/"no" responses, entity names or similar short expressions, or lists of entity names and similar short expressions; and there are no "exact" answers in the case of summary questions. The maximum allowed length of each "ideal" answer is 200 words.

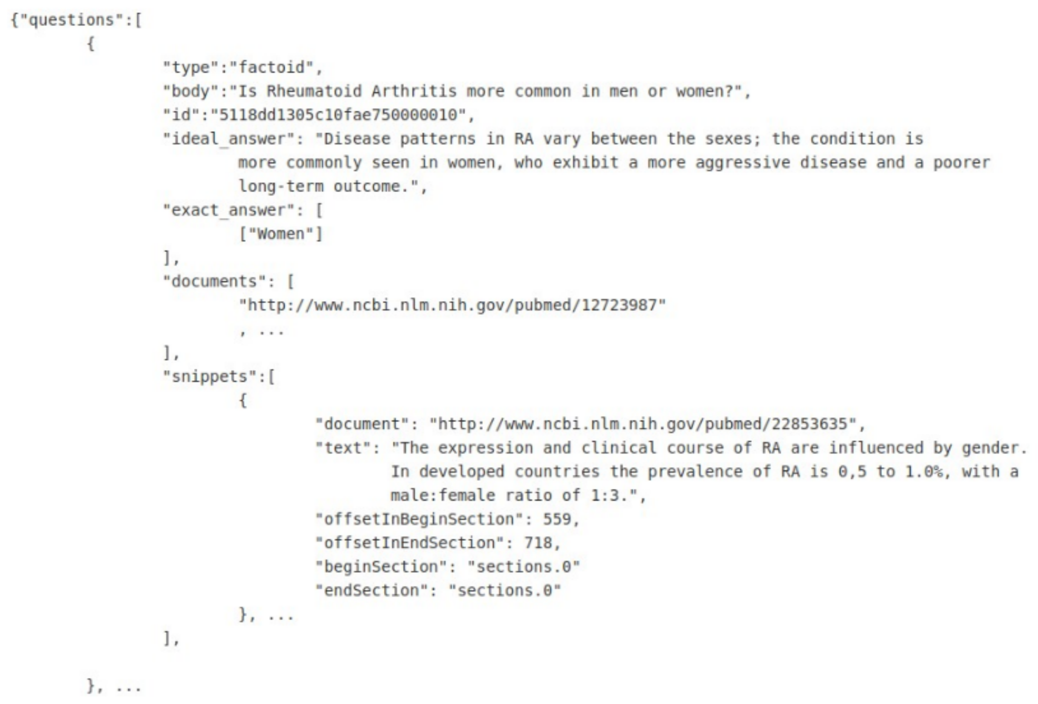
5.2. The main challenges inherent in this task are:

- Accurately retrieving semantically relevant documents beyond simple keyword matching
- Extracting precise factual entities or lists from complex biomedical text
- Generating coherent and comprehensive ideal answers (summaries) from multiple documents

This work aims to automate these tasks by integrating modern Natural Language Processing (NLP) techniques, thereby improving the efficiency, consistency, and quality of biomedical information retrieval and question answering, ultimately supporting biomedical research and clinical decision-making.

5.3. JSON Format Of Data

Figure 1 gives the json format of the data. This format is used in the training dataset provided to us. For the answers to be submitted, this format had to be used with only those labels that are required for answers of that phase.



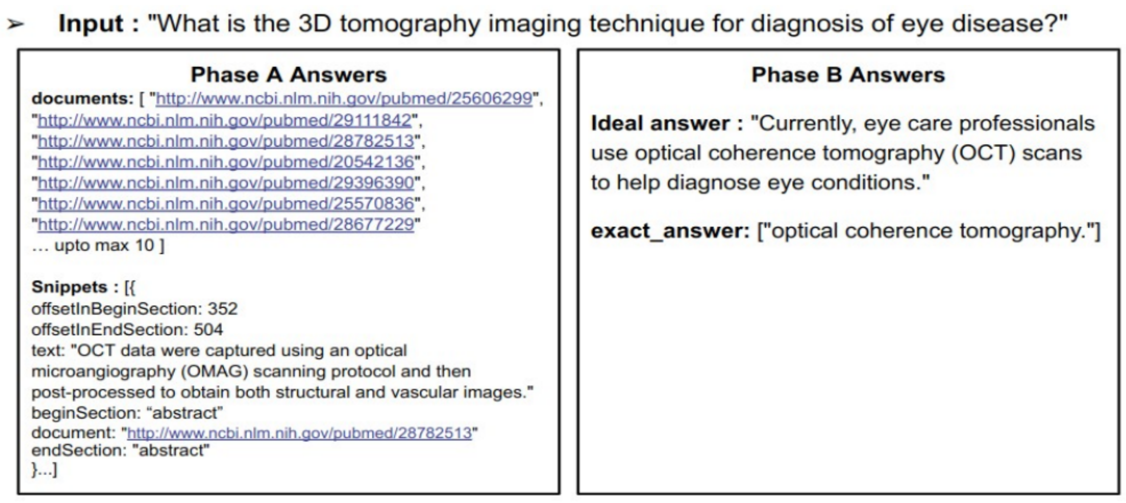
```
{"questions":[
        {
                "type":"factoid",
                "body":"Is Rheumatoid Arthritis more common in men or women?",
                "id":"5118dd1305c10fae750000010",
                "ideal_answer": "Disease patterns in RA vary between the sexes; the condition is
                        more commonly seen in women, who exhibit a more aggressive disease and a poorer
                        long-term outcome.",
                "exact_answer": [
                        ["Women"]
                ],
                "documents": [
                        "http://www.ncbi.nlm.nih.gov/pubmed/12723987"
                        , ...
                ],
                "snippets":[
                        {
                                "document": "http://www.ncbi.nlm.nih.gov/pubmed/22853635",
                                "text": "The expression and clinical course of RA are influenced by gender.
                                        In developed countries the prevalence of RA is 0,5 to 1.0%, with a
                                        male:female ratio of 1:3.",
                                "offsetInBeginSection": 559,
                                "offsetInEndSection": 718,
                                "beginSection": "sections.0"
                                "endSection": "sections.0"
                        }, ...
                ],

        }, ...
```

Figure 1: JSON format of data

5.4. Input Output Examples

Figure 2 shows an example of input, i.e a question and corresponding answer for each phase.



➢ **Input :** "What is the 3D tomography imaging technique for diagnosis of eye disease?"

Phase A Answers

documents: ["http://www.ncbi.nlm.nih.gov/pubmed/25606299", "http://www.ncbi.nlm.nih.gov/pubmed/29111842", "http://www.ncbi.nlm.nih.gov/pubmed/28782513", "http://www.ncbi.nlm.nih.gov/pubmed/20542136", "http://www.ncbi.nlm.nih.gov/pubmed/29396390", "http://www.ncbi.nlm.nih.gov/pubmed/25570836", "http://www.ncbi.nlm.nih.gov/pubmed/28677229"
... upto max 10]

Snippets : [{
offsetInBeginSection: 352
offsetInEndSection: 504
text: "OCT data were captured using an optical microangiography (OMAG) scanning protocol and then post-processed to obtain both structural and vascular images."
beginSection: "abstract"
document: "http://www.ncbi.nlm.nih.gov/pubmed/28782513"
endSection: "abstract"
}...]

Phase B Answers

Ideal answer : "Currently, eye care professionals use optical coherence tomography (OCT) scans to help diagnose eye conditions."

exact_answer: ["optical coherence tomography."]

Figure 2: Example of Input and Answers

6. Proposed Approach

The main task can be easily divided into 2 Systems (or pipelines) . One system for Phase A, and one for Phase A+ and Phase B. Phase A+ problem can be easily converted into a Phase B problem. In Phase B we are given gold documents and snippets related to each question and we have to return exact and ideal answers to these questions. In Phase A+ problem we are given the questions only and no gold documents or snippets, So we can use System 1 to get documents and snippets for each question and thus we can use System 2 to get our answers for Phase A+. Since Test Data in Phase A+ is same as the test data in phase A, therefore we can easily use the answers of Phase A Test data from System A and use System 2 on it to get the Phase A+ answers. Thus we only require 2 System for the whole project.

The whole project can be divided into 5 Subtasks for a structured approach.
Relevant Document Retrieval=>Subtask 1
Ranking Documents=>Subtask 2
Snippets Extraction=>Subtask 3
Extracting Exact Answers=>Subtask 4
Generating Ideal Answers (Summarization)=>Subtask 5
System 1 solves the Subtask 1,2 and 3
System 2 solves the Subtask 4 and 5

6.1. Approach For Each Subtask

6.1.1. Subtask 1 – Document Retrieval

In this subtask the main goal is to retrieve relevant documents related to the given question. According to our approach the question can be transformed into a query consisting of important keywords (say Nouns). Now this query is used to fetch and retrieve the related documents. The method used by us to form the query is discussed in the subsection 7.3 (System 1 implementation) of the section 7 (Implementation).

To retrieve relevant documents, we utilize the Entrez Programming Utilities (E-utilities)[4] — a suite of web-based APIs provided by the National Center for Biotechnology Information (NCBI)[5] for programmatic access to the PubMed database.

The query formed is then submitted to Entrez using the following utilities:

- esearch: Retrieves PubMed IDs (PMIDs) of articles(documents) matching the query.
- efetch: Fetches metadata and abstracts of the retrieved articles(articles) based on the PMIDs.

Entrez is used as it offers direct and authoritative access to the entire PubMed database, making it ideal for biomedical information retrieval. It enables automated, efficient, and large- scale querying, ensuring comprehensive coverage of relevant literature while maintaining scalability and reliability in a production system.

6.1.2. Subtask 2 – Document Ranking

Subtask 2 deals with ranking a given set of documents based on a question provided. It then returns a list back where the list is sorted in a descending order of relevance.

To accurately rank retrieved documents based on their relevance to a question, we employ the sentence-transformers/all-MiniLM-L6-v2[6] model — a lightweight and efficient transformer- based model that captures semantic similarity between text pairs, going beyond simple keyword matching.

Given a biomedical question and the abstract of an article, our fine-tuned model encodes both into fixed-size vector representations (embeddings). These embeddings capture the contextual meaning of the text, allowing us to compute their cosine similarity and rank documents by their semantic closeness to the question.

Model Architecture: The model is based on a distilled Transformer architecture (MiniLM) — a compact and faster variant of BERT[7]. Key characteristics include:

[4]https://www.ncbi.nlm.nih.gov/books/NBK25497/
[5]https://www.ncbi.nlm.nih.gov/
[6]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
[7]https://huggingface.co/docs/transformers/en/model_doc/bert

- 6 Transformer layers
- 384-dimensional embeddings
- Generates dense, fixed-size vectors suitable for fast similarity computation.

The main reason to choose MiniLM was that it enables context-aware document ranking that captures semantic relationships between questions and abstracts, making it more accurate and robust, all while remaining computationally efficient for large-scale use.

The model used will be fine-tuned using the Multiple Negatives Ranking Loss on a large corpus of sentence pairs. This objective trains the model to embed semantically similar sentences closer together in the embedding space, improving its ability to assess relevance effectively.

6.1.3. Subtask 3 – Extracting Snippets

In this Subtask, the snippet extraction part is done. Snippets are extracted from the Abstracts of the retrieved documents. Here, a pre-trained Extractive Question-Answering Model is used. No fine-tuned model is used here.

We utilize the deepset/roberta-base-squad2[8] model — a robust question answering (QA) model capable of identifying exact answer spans within a given context.

This model is based on RoBERTa-base[9] - (Robustly Optimized BERT Pretraining Approach) and is fine-tuned on the SQuAD 2.0[10] dataset, which enables it to:

- Extract concise answer spans from the provided text when a valid answer exists.
- Abstain from answering gracefully when no appropriate answer is found in the context.

For each biomedical question, the abstract of a retrieved document serves as the context. The QA model takes the question-context pair as input and predicts the start and end positions of the most relevant answer span within the abstract. If no suitable answer is detected, the model returns a null output, minimizing false positives. 5 snippets are extracted using this model and then the one with the best score and length combination is selected out of these.

Model Architecture and Capabilities

- Built on the RoBERTa-base transformer architecture (12 layers, 768 hidden size).
- Fine-tuned explicitly for extractive question answering with a "no answer" option (a key feature of SQuAD 2.0).
- Capable of both precise extraction and answerability detection, making it well-suited for biomedical contexts where relevant information may not always be explicitly present.

6.1.4. Subtask 4 – Extracting Exact Answers

For generating concise, fact-based exact answers, we employ the T5 (Text-to-Text Transfer Transformer) — a flexible Transformer-based encoder-decoder model that frames all NLP tasks as a text-to-text problem. A model - (google-t5/t5-base) [11] is finetuned by us for the task.

The input to the model consists of:

- The biomedical question
- Its question type (e.g., yes/no, factoid, list)
- A concatenation of all answer snippets from the relevant documents

[8]https://huggingface.co/deepset/roberta-base-squad2
[9]https://huggingface.co/FacebookAI/roberta-base
[10]https://huggingface.co/datasets/rajpurkar/squad_v2
[11]https://huggingface.co/google-t5/t5-base

The T5 model will be fine-tuned for this exact answer task. The training dataset is, around $\tilde{5}300$ questions. The model is fine-tuned by providing the required input and output format. This combination makes the model learn the pattern of extracting the exact answer from a given output.

The output given must be post-processed according to each of different question types mentioned as each of them requires a different type of exact answer.

T5 is specifically used because its flexible text-to-text architecture allows explicit control over output formats based on question type, improving exact answer accuracy across diverse biomedical questions.

6.1.5. Subtask 5 – Generating Ideal Aanswers

For generating ideal answers — comprehensive, well-formed summaries that synthesize information from multiple snippets — we again leverage the T5 model, fine-tuned for generative summarization.

The input to the model includes:

- The biomedical question
- All extracted snippets concatenated into a single context

The model generates a coherent, paragraph-style summary that integrates relevant information providing a richer, more contextualized answer to the question.

Again, the model (google-t5/t5-base) is fine-tuned on a large set of question-answer pairs, around $\tilde{5}300$ questions, which enables it to learn the patterns required for ideal answer generation. For fine-tuning the input is the question along with all the relevant snippets as context and the output as it was in dataset. This fine-tuning helps it to generate long-form, informative summaries aligned to our needs.

T5 is used because it's encoder-decoder framework and large pretrained knowledge base make it ideal for multi-document summarization, providing comprehensive and context-aware answers.

These Subtasks are solved by the Pipeline System 1 and System 2. The division was made to easily understand all the subtasks that need to be done in the main task.

7. Implementation

The approach is implemented in the following manner :-

7.1. Pre-Processing

The Training (or Development) Dataset provided to us contains questions along with related documents, answers, but no abstract of documents is given. For fine-tuning we need the abstract part as the document label contain just the link itself. Thus for each question, we need to extract the related abstracts. But the issue is the limit of Entrez, which is used to extract abstracts using efetch. Entrez has a limit of fetching 3 documents per second. Thus for such a large dataset we cannot extract all abstracts in a single run as it would take many many hours.

Thus the dataset is firstly divided into 6 parts with first 5 having 1000 questions and last one having 300 questions. Now for each of these parts abstracts are extracted using Entrez. At last abstracts from all these parts are appended to the main dataset. Thus creating a solid training dataset will all data related for fine-tuning.

7.2. Finetuning

For Subtask 2, 4 and 5 the models which will be used needs to be fine-tuned first. Each of the model is fune-tuned as per it was described in the approach above. Models are fine-tuned in Google Colab as our current machine cannot perform computation and GPU intensive tasks like fine-tuning.

The fine-tuning part was time-consuming too as one needs to change the training arguments such as learning-rate, number of epochs, etc to get the perfect fine-tuned model that gives the best answers.

7.3. System 1 Implementation :-

Figure 3 shows the framework of System 1.

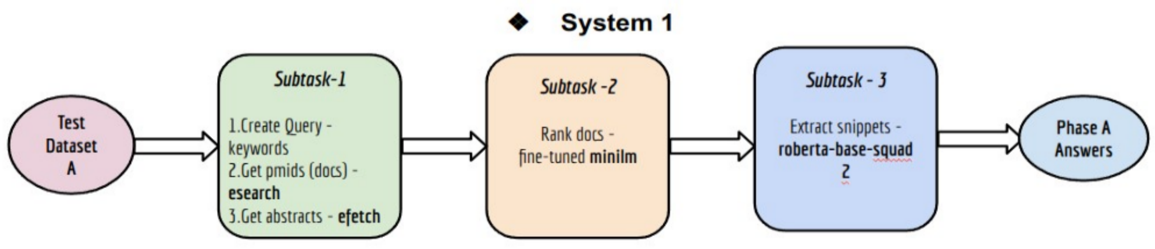

System 1
Test Dataset A
Subtask-1
1.Create Query - keywords
2.Get pmids (docs) - esearch
3.Get abstracts - efetch
Subtask -2
Rank docs - fine-tuned minilm
Subtask - 3
Extract snippets - roberta-base-squad 2
Phase A Answers

Figure 3: System 1 Framework

The system 1 contains a python function that accepts a Test Json and Email and return the answer data in the required submission format. Test Json is the file that contains questions given to us for whose the answers need to be given, and Email is the email that is used for Entrez login. The model used here is minilm_ranked (our fine-tuned MiniLM) and is stored in the respective directory to use it. This model is then loaded inside this function, to be used for article (document) ranking. The model used for extracting snippet is the extractive question-answering model “deepset/roberta-base-squad2” which is taken from huggingface. This model is loaded inside this function using pipeline module from transformers library, for the use of snippet extraction from relevant articles(documents).

Now inside this function, two functions are defined, each of which takes a question. One returns nouns, proper nouns and verbs separated by commas while other returns only nouns and proper nouns. This step assists in the query creation process.

Now to get the relevant articles for each question we create the query using above method and then for each query we get PMIDS of relevant articles using Entrez.esearch . At first nouns, proper nouns along with verbs are searched as query and if no articles found then, only nouns and proper nouns are searched as query. If then also no PMIDs or articles are found then the whole question is searched using Entrez.esearch . After getting the PMIDs the document link is created from these PMIDs and appended to answer data. This completes our article (document) retrieval process.

After that for ranking the articles (documents) we need to have the abstract of each of document because we cannot rank articles based on just article links. So, for every PMID, abstracts are fetched using Entrez.efetch . Then the minilm_ranker model (fine-tuned MiniLM) is loaded. For each question, we pass the question to a function that takes the ranking model, article abstracts and PMIDs as input. Now the function ranks the documents according to relevance and returns a list containing PMIDS, articles in a descending order of relevance. PMIDs are used so to identify the article link. Now these ranked article (documents) are appended to question data in the dictionary replacing the original unranked ones. This dictionary itself is returned at last after all the steps giving us the answer data in required format of submission.

After this the snippet extraction part is done. For that we use deepset/roberta-base-squad2 model from hugging face. Then we create a QA pipeline using pipeline from transformers. Now every question is passed in QA format to this pipeline to extract the answer snippet. Input is question + abstract of an article. Output is the snippet extracted. All other details like offsetInBeginSection, offsetInEndSection,etc are filled from start_pos, end_pos,etc. Document(article) link is also appended, along with beginSection and endSection which are abstract. Snippets, thus extracted, are appended to the answer data.

At last the answer data is returned. This data contains the articles (documents) in a ranked order of decreasing relevance, along with snippet text from each article for every question.

Now, this answer data is in a python dictionary format. The data is then converted to a .json file format as required for the submission.

7.4. System 2 Implementation :-

Figure 4 shows the framework of System 1.

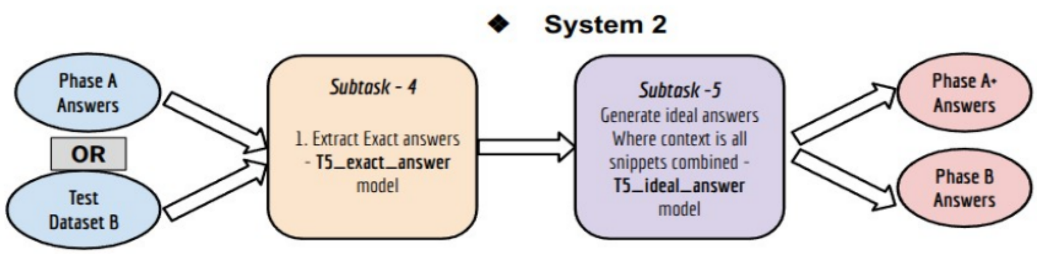

System 2
Phase A Answers
OR
Test Dataset B
Subtask - 4
1. Extract Exact answers - T5_exact_answer model
Subtask -5
Generate ideal answers Where context is all snippets combined - T5_ideal_answer model
Phase A+ Answers
Phase B Answers

Figure 4: System 2 framework

The System 2 contains a python function that accepts only the .json file in which questions are given, along with gold articles and snippets, whose exact and ideal answers are required. The models used here are the fine-tuned T5 sentence transformers :- T5_ideal_answer and T5_exact_answer. These models need to upload to the required directories so that they can be loaded inside the function.

Inside this function, two functions named get_ideal_answer and get_exact_answer are defined. The get_exact_answer function takes question, question type, its snippets, model and tokenizer as inputs. It uses the question, question type and snippets to form an input and then get the exact answer from the input. Question, question type and snippets are all appended to form the input. Since T5 is a text-to-text model, so the exact answer extracted is in string format. But exact answer for list and factoid type question must be list and for yes/no type it should be "yes" or "no" and no exact answer is to be given for summary type. So according to each question type, the post- processing is done and required format of exact answer is taken from answer string. The get_ideal_answer takes the question, its snippets, model and tokenizer as input and generate a ideal answer for the question using all the snippets as the context. The ideal answer is a summarized answer for the question based on the snippets related to the question.

For every question these functions are run to get ideal and exact answers which are then appended to the answer data which is to be returned by the main function. This answer data is in a python dictionary format. The data is then converted to a .json file format as required for the submission.

8. Results

Participated in shared task BioASQ Task13b, Test Batch 4. These are the official results from their website.

Manual Evaluation has not been done yet. These are auto-generated results from evaluation script/s. Various types of metrics have been used. Each phase with its different required answers, uses a set of different metric. The result along with the metrics used has been provided in a tabular form below.

8.1. Phase A Results

The results of Phase A : Documents are in the Table 1.

The results of Phase A : Snippets are in the Table 2.

8.2. Phase B Results

The results of Phase B : Exact Answers are in the Table 3.

The results of Phase B : Ideal Answers are in the Table 4.

8.3. Phase A+ Results

The results of Phase A+ : Exact Answers are in the Table 5.
The results of Phase A+ : Ideal Answers are in the Table 6.

Table 1
Phase A Documents Result.

System	Mean precision	Recall	F-measure	MAP	GMAP
1. PhaseA_System	0.0154	0.0346	0.0202	0.0291	0.0000

Table 2
Phase A Snippet Result.

System	Mean precision	Recall	F-measure	MAP	GMAP
1. PhaseA_System	0.0068	0.0138	0.0080	0.0125	0.0000

Table 3
Phase B exact answer result

	YES/NO				Factoid			List		
System	Accu-racy	F1 yes	F1 No	Macro F1	Strict Acc.	Lenient Acc.	MRR	Mean Prec.	Recall	F-measure
3.PhaseB_System	0.7308	0.8444	-	0.4222	0.1818	0.1818	0.1818	0.0531	0.0563	0.0536

Table 4
Phase B Ideal Answer Result

	Automatic Scores(Rouge-R)				Manual Scores
System	R-2(Rec)	R-2(F1)	R-SU4(Rec)	R-SU4(F1)	Not Given Yet
3.PhaseB_System	0.3137	0.3336	0.3109	0.3293	Not Given Yet

Table 5
Phase A+ exact answer result

	YES/NO				Factoid			List		
System	Accu-racy	F1 yes	F1 No	Macro F1	Strict Acc.	Lenient Acc.	MRR	Mean Prec.	Recall	F-measure
3.PhaseB_System	0.7308	0.8444	-	0.4222	0.1364	0.1364	0.1364	-	-	-

Table 6
Phase A+ Ideal Answer Result

System	Automatic Scores (Rouge-R)				Manual Scores
	R-2(Rec)	R-2(F1)	R-SU4(Rec)	R-SU4(F1)	
3.PhaseB_System	0.0826	0.0901	0.0920	0.0986	Not Given Yet

9. Discussions

The current systems presents a functional baseline that integrates all required subtasks—document retrieval, ranking, snippet extraction, and answer extraction and generation. While the overall results are not very good, they do validate the feasibility of the modular approach. Each component/Subtask in the system, though individually limited in performance, demonstrates that the intended function is operational. This confirms that the foundational design of the system is sound and capable of supporting future improvements.

The underwhelming results highlight the need for significant enhancements in several areas. Some subtasks, particularly document ranking and answer extraction, require more robust techniques to capture biomedical relevance and generate high-quality responses. However, the current framework is flexible and modular, allowing for easy substitution or augmentation of individual components. This means that alternative models, especially those more specialized or fine-tuned for biomedical domains, can be tested and incorporated without requiring an overhaul of the entire system.

In essence, the baseline system successfully establishes a starting point. It confirms that the problem is addressable using current tools and provides a clear structure for future experimentation. The path forward involves systematic testing of alternative models, targeted fine-tuning, and iterative evaluation to gradually replace weaker components and improve the overall performance.

10. Conclusion And Future Work

This work presents Biomedical Question Answering (QA) systems that integrates information retrieval (IR), extractive QA, and abstractive summarization to handle a variety of biomedical question types. The systems employs a combination of MiniLM (for semantic document ranking), RoBERTa_SQuAD2 (for answer span extraction), and T5 (for both exact and ideal answer generation). Despite the limited performance of the current approach, the system demonstrates that such a modular and semantically driven architecture is viable and provides a foundation for further research in biomedical information access.

The results indicate that the baseline is functional but lacks the robustness needed for high-quality biomedical QA. Several components, particularly in document ranking and long-form answer generation, require further refinement.

10.1. Future Work :-

To improve system performance and answer quality, the future work can incorporate the following points:

- Model Improvement : Experiment with a broader range of pretrained models, especially those fine-tuned specifically on biomedical corpora, to enhance domain- specific understanding.
- Relevance Optimization: Develop advanced document ranking strategies that consider domain-specific signals and query intent to boost retrieval precision.
- Enhanced Summarization: Improve the abstractive summarization component by fine-tuning on biomedical summarization datasets or using models designed for medical discourse generation.
- System Improvement: Conduct error analysis across subtasks to identify weaknesses and iteratively improve individual modules without compromising the overall system structure.

By addressing these, the baseline systems can evolve into a more accurate and reliable biomedical QA solution capable of supporting clinicians, researchers, and end-users in accessing critical medical information.

Acknowledgments

Thanks to the developers of ACM consolidated LaTeX styles https://github.com/borisveytsman/acmart and to the developers of Elsevier updated LaTeX templates https://www.ctan.org/tex-archive/macros/latex/contrib/els-cas-templates.

Declaration on Generative AI

During the preparation of this work, the author(s) used Chat-GPT-4 in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] A. Nentidis, G. Katsimpras, A. Krithara, M. Krallinger, M. Rodríguez-Ortega, E. Rodriguez-López, N. Loukachevitch, A. Sakhovskiy, E. Tutubalina, D. Dimitriadis, G. Tsoumakas, G. Giannakoulas, A. Bekiaridou, A. Samaras, F. N. Maria Di Nunzio, Giorgio, S. Marchesin, M. Martinelli, G. Silvello, G. Paliouras, Overview of BioASQ 2025: The thirteenth BioASQ challenge on large-scale biomedical semantic indexing and question answering, in: L. P. A. G. S. d. H. J. M. F. P. P. R. D. S. G. F. N. F. Jorge Carrillo-de Albornoz, Julio Gonzalo (Ed.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Sixteenth International Conference of the CLEF Association (CLEF 2025), 2025.

[2] A. Nentidis, G. Katsimpras, A. Krithara, G. Paliouras, Overview of BioASQ Tasks 13b and Synergy13 in CLEF2025, in: G. Faggioli, N. Ferro, P. Rosso, D. Spina (Eds.), CLEF 2025 Working Notes, 2025.

[3] A. Nentidis, G. Katsimpras, A. Krithara, S. Lima López, E. Farré-Maduell, L. Gasco, M. Krallinger, G. Paliouras, Overview of BioASQ 2023: The Eleventh BioASQ Challenge on Large-Scale Biomedical Semantic Indexing and Question Answering, Springer Nature Switzerland, 2023, p. 227–250. URL: http://dx.doi.org/10.1007/978-3-031-42448-9_19. doi:10.1007/978-3-031-42448-9_19.

[4] A. Krithara, A. Nentidis, K. Bougiatiotis, G. Paliouras, BioASQ-QA: A manually curated corpus for Biomedical Question Answering, Scientific Data 10 (2023) 170. URL: https://doi.org/10.1038/s41597-023-02068-4. doi:doi.org/10.1038/s41597-023-02068-4.

[5] A. Krithara, J. G. Mork, A. Nentidis, G. Paliouras, The road from manual to automatic semantic indexing of biomedical literature: a 10 years journey, Frontiers in Research Metrics and Analytics Volume 8 - 2023 (2023). URL: https://www.frontiersin.org/journals/research-metrics-and-analytics/articles/10.3389/frma.2023.1250930. doi:10.3389/frma.2023.1250930.

[6] S. Robertson, H. Zaragoza, The probabilistic relevance framework: Bm25 and beyond, Foundations and Trends® in Information Retrieval 3 (2009) 333–389. URL: http://dx.doi.org/10.1561/1500000019. doi:10.1561/1500000019.

[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL: https://arxiv.org/abs/1810.04805. arXiv:1810.04805.

[8] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, 2019. URL: https://arxiv.org/abs/1907.11692. arXiv:1907.11692.

[9] P. Rajpurkar, R. Jia, P. Liang, Know what you don't know: Unanswerable questions for squad, 2018. URL: https://arxiv.org/abs/1806.03822. arXiv:1806.03822.

[10] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, M. Zhou, Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. URL: https://arxiv.org/abs/2002.10957. arXiv:2002.10957.

[11] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.