This MICCAI paper is the Open Access version, provided by the MICCAI Society. It is identical to the accepted version, except for the format and this watermark; the final published version is available on SpringerLink.

# MedDocBench: A Benchmark for Medical Document Visual Question Answering Supplementary Material

J Snehan✉ and Kunal Singh

Fractal Analytics
snehan2k2@gmail.com

## S1 Benchmark Question Examples

Figure S1 illustrates the four task types on a single multi-page discharge summary from MedDocBench, with one representative question, its gold answer, and the supporting evidence or reasoning for each.

## S2 Additional Failure Modes

The Error Analysis section of the main paper details three failure modes and summarizes two more. We describe those two in full here.

**Dense, Wide, Multi-Column Panels.** Allergy panels and blood-count panels list many tests across several columns on a single page, often tens of tests at once. In practice these are read exhaustively, and every row is checked. Models instead read only part of the panel and skip the remaining rows. When asked to count or to list all the tests that match a condition, they therefore return too few. General documents rarely pack this many small rows across the full width of a page.

**Fabrication on False Clinical Premises.** Answerability questions ask for a fact that is not in the report. The safe answer is to say the fact is not present. Instead, models often invent a value that looks clinically reasonable. Examples include a lab result for a date on which it was never measured, a reference cutoff that is not printed, and a dose worked out from a body weight that is not given. A made-up drug dose is worse than saying the answer is not available. This is the most dangerous failure mode.




**Fig. S1.** The top strip shows the report pages. Each panel gives a representative question, its gold answer, and a cropped snippet of the report evidence that grounds it.

## S3 Automated Quality Verification

The synthesized pairs are checked by an automated verifier that filters out low-quality items. A judge LLM reads the page (or the whole document, for cross-page and unanswerable items) along with the generated questions, and marks each one to keep or drop with a short reason. All questions for a page or document are judged together in a single call. Verification was performed with Grok-4.3, a model from a different family than the one used for synthesis, which makes the

check complementary, so that biases of the generating model are less likely to pass unnoticed.

Each item is judged on three criteria and kept only if all pass. **Grounding** checks that the fields, rows, values, or entities the question refers to actually appear in the document, so it does not rely on outside knowledge or on OCR errors. **Answer shape** checks that the answer matches its declared answer type, that a list is used only when the answer is really several items, and that the answer stays a single clean value with no sentences or units. **Task label** checks that the assigned sub-type fits the question and that the item belongs to its task at all; for example, an understanding item must be a real lookup and not hidden reasoning, and an out-of-range question is dropped unless a reference range is printed on the page. For cross-page items the verifier also confirms that the question really needs its cited pages.

## S4 Fairness of the Evaluation Panel

GPT-5 for synthesis and Grok-4.3 for verification also appear in the evaluated panel, but neither role confers a direct advantage. Synthesis only proposes questions whose gold answers are grounded in the document and then checked and corrected by human annotators, verification only accepts or rejects items without authoring any answer, and the Surya OCR reused by the OCR+LLM baselines only transcribes page text and never produces answers.

## S5 Evaluation Prompts

Each query combines a task-specific system prompt with a user message that carries the page image(s) and the question. The system prompt for each of the four task types is given below.

### Single-Page Understanding

```
Answer the question using only the document page image.
Give just the value(s) exactly as written on the page, with
no field names, labels, or extra words.
Reply with ONLY a JSON object: {"answer": <value>}, where
<value> is a single value (a string or a number) for one
answer, or a JSON array for several answers. Output nothing else.
```

### Single-Page Reasoning

```
Answer the question using only the document page image.
Give only the final answer; for a number, output the bare
number with no units or extra words.
Reply with ONLY a JSON object: {"answer": <value>}, where
<value> is a single value (a string or a number) for one
answer, or a JSON array for several answers. Output nothing else.
```

### Cross-Page Reasoning

```
Answer the question using only the document page images
(each marked '=== PAGE: <id> ===').
Give only the final answer; for a number, output the bare
number with no units or extra words.
Reply with ONLY a JSON object: {"answer": <value>}, where
<value> is a single value (a string or a number) for one
answer, or a JSON array for several answers. Output nothing else.
```

### Answerability

```
Answer the question using only the document page images
(each marked '=== PAGE: <id> ===').
Give only the final answer; for a number, output the bare
number with no units or extra words.
If the document does not contain the answer, reply with
{"answer": "N/A"}.
Reply with ONLY a JSON object: {"answer": <value>}, where
<value> is a single value (a string or a number) for one
answer, or a JSON array for several answers. Output nothing else.
```

The user message pairs the image content with the question. Single-page tasks use one page image. Cross-page and Answerability tasks include every page, each preceded by a delimiter.

### User Message (Single-Page)

```
[page image]
QUESTION: <question>
```

### User Message (Multi-Page)

```
=== PAGE: <id> ===
[page image]
... (repeated for every page)
QUESTION: <question>
```

## S6 Data Generation Prompts

The prompts driving the construction pipeline are summarized below. Each generator and the verifier receive the page image as the authoritative source together with its Surya OCR markdown as a reading aid, and return a fixed schema.

## S6.1 Grounded QA Synthesis

### Single-Page Understanding Generator

```
You are an expert annotator building single-page DocVQA data. You are
given the page IMAGE (authoritative ground truth) and a MARKDOWN OCR
transcription as a reading aid; when they disagree, trust the image.

Write understanding QA pairs, grounded in the image, that test reading
and extraction. Cover the types: key_value_extraction,
table_cell_extraction, table_row_lookup, list_extraction, and
multi_field_extraction.

Rules: every question is answerable from this page alone and has exactly
one unambiguous answer. Copy the answer STRICTLY VERBATIM as printed
(units, casing, even OCR errors); never compute or paraphrase. Provide a
verbatim evidence snippet containing every answer value. Answers must be
atomic and exact-match verifiable (ANLS); a unit is never an answer. Skip
placeholder / PII fields and anything not literally on the page.

Coverage: at most one question per type (<= 5 total); emit fewer, or
none, rather than contrived questions, and keep each on a distinct fact.
A type is applicable only if the page truly contains that content; never
fabricate a table or list.
```

### Single-Page Reasoning Generator

```
You are an expert annotator building single-page DocVQA REASONING data.
Inputs are the page IMAGE (authoritative) and an OCR MARKDOWN aid; trust
the image on conflict.

Write questions that REQUIRE reasoning over the page, not a direct
lookup. Cover the types: multi_hop, conditional_filter, out_of_range,
comparison, counting, indirect_reference, and negation.

Rules: answerable from this page only. Each question needs a genuine
derivation (filter, range-check, compare, count, chain, negate); reject
single-cell lookups. Derived answers need not appear verbatim but must be
uniquely determined. Answers are atomic and exact-match verifiable (ANLS)
-- a bare number or a short label -- with all explanation placed in
`reasoning`, which must cite the exact page values used. Do not combine a
direction and a magnitude in one question, and use only reference
intervals printed on the page.

Coverage: at most one question per type (<= 7 total); skip a type rather
than fabricate data, and keep every question on a distinct fact.
```

### Cross-Page Reasoning Generator

```
You are an expert annotator building CROSS-PAGE DocVQA reasoning data for
an entire multi-page report. Each page is delimited as "=== PAGE: <id>
===" with its image (authoritative) and OCR markdown aid; track which
fact lives on which page.

Write questions that COMBINE information from two or more pages and
require reasoning; discard anything answerable from a single page or by a
direct lookup, and prefer questions spanning three or more pages when the
report allows. Label each with one cross_page_link: temporal_trend,
event_effect, range_transfer, aggregation, find_then_fetch, ranking,
spanning_filter, or summary_detail.

Rules: at least two distinct pages, recorded in `source_pages`; do not
reference pages by number in the question text. Answers are atomic and
exact-match verifiable, with the derivation and per-page values in
`reasoning`. No fake cross-page (two stapled lookups), no forced
linkages, no near-duplicates, printed ranges only.

Coverage: 5 is a hard ceiling, not a goal; most reports support only 1--3
genuine questions. Count the independent cross-page relationships first
and emit at most that many; prefer variety across link types and emit 0
rather than padding.
```

### Answerability Generator

```
You are an expert annotator building UNANSWERABLE DocVQA items for
negative testing (whether a model correctly abstains with "N/A"). Inputs
are the document pages (images authoritative, OCR markdown aid).

Write questions that READ like genuine questions grounded in real anchors
(sections, tests, dates that actually appear), yet whose final needed
fact is ABSENT from the document; the answer is always "N/A".

The bar: a question is unanswerable only if the document yields NO
definitive answer -- a concrete negative such as "none" / "0" / "not
reported" makes it ANSWERABLE, so drop it. Plausibility is mandatory: a
grader must not be able to tell it is unanswerable from the wording
alone, and never signal it with "if any" / "if reported". Only the final
fact is missing; never invent anchors. Name the missing fact in
`missing_information` and justify the gap in `reasoning`.

Coverage: at most 3; most documents support only 1--2. Prefer fewer, or
0, over contrived or actually-answerable questions.
```

## S6.2 Automated Quality Verification

### Quality Verifier

```
You are a strict quality-control verifier for a DocVQA dataset. You are
given the page IMAGE (authoritative), an OCR MARKDOWN aid, and the
numbered QA items generated for the page; for each item decide KEEP or
DROP.

Do NOT check answer correctness -- assume the answer value is correct and
never drop an item for a value you disagree with. Judge only three
criteria, keeping an item only if all pass:
1. Grounding: the fields, rows, values, or entities referenced actually
appear on the page; drop if it needs outside knowledge or rests on OCR
noise.
2. Answer shape: answer_type matches the value; a list is used only for
genuinely multiple items; the answer is atomic and clean (no sentences,
no units on numbers).
3. Task label: the question_type fits the question and the item truly
belongs to its task.

Return one keep/drop verdict per item.
```

### Answerability Verifier (Second Pass)

```
You are a strict DocVQA solver used to VALIDATE supposedly unanswerable
questions. Given the document (images authoritative, OCR markdown aid)
and a list of questions, genuinely TRY to answer each using only the
document. Mark 'answerable' -- and record the concrete answer, including
a negative such as "none" / "0" -- if the document yields or implies any
definitive answer; mark 'not_answerable' (answer "N/A") only if the
needed fact is truly absent and not derivable. Be skeptical and literal.
```

Model-based difficulty filtering does not use a new prompt. Each surviving question is answered by four small vision models using the evaluation prompts in Section S5, and is dropped only if all four exactly match the gold answer.