

A Cost-Efficient Vision-Text Cascade for Document QA: What Resolution Buys and What It Doesn't

Urjita Bedekar

Independent Researcher

urjita9@gmail.com

Abstract

This paper presents a two-stage vision-text cascade, submitted to the DocSem shared task at DocInsights Workshop, EMNLP 2026. A cheaper vision model (Claude Haiku 4.5) locates and transcribes the block that answers the question, and a stronger text-only model (Claude Sonnet 4.5) turns it into an arithmetic expression, which Python evaluates. Keeping the expensive model off the page images reduces cost by 14% against using one model for both, and the system reaches 88.02% answer accuracy on validation, up from 82.95%. On the test split the same system scores 7.28%, and this paper diagnoses the difference. The main constraint is block selection: test documents contain decoy blocks that repeat the question's words while asking something else. Prose inside the blocks is transcribed almost perfectly; identifiers, which carry no redundancy, run at 75–81%. I also find that the two splits are stored at different resolutions, 220 dpi against 71–79 dpi, and that enlarging a low-resolution page before sending it improves identifier transcription by 58 points. The gain stops at the provider's 1568 px cap, above which rendering higher delivers an identical image and accuracy does not meaningfully improve.

1 Introduction

Recent advances in Natural Language Processing for Information Retrieval have given rise to document-based question answering. This is an important case, as much of the data exists in documents. Document understanding and question answering over document images has shown remarkable progress (Mathew et al., 2021). The purpose of this system is to return an answer along with a citation to the passage it came from, so a reader can check it (Rashkin et al., 2023). This can be used in a wide variety of applications including but not limited to clinician notes, meeting notes, or product-specific information for an enterprise chatbot.

It is a common assumption that a system's behaviour on the test set is reliably predicted by its behaviour on the validation set. This did not hold here. The paper proposes a cost-efficient system which reached 88.02% accuracy on the validation set, up from 82.95% for a single model. The cost-efficient approach used here, an LLM cascade, is described by Chen et al. (2023). Evidence F1 reached 1.00 for 62 of the 69 teams on the validation leaderboard, while answer accuracy across those same systems ranged from 23% to 100%, so the attribution metric separates almost nobody. OCR (Smith, 2007) was also studied, but the watermark contaminated the extracted text and only 21% of identifier tokens were recovered correctly. This makes a VLM the more practical choice for a corpus whose PDFs contain no extractable text layer. To process multi-page image documents under strict resource constraints, the system uses a two-stage vision-text cascade. Stage 1 uses a lightweight, cheaper VLM to locate and transcribe blocks from the PDF. Stage 2 passes the resulting text tokens, and zero image tokens, to a stronger model which returns a formal arithmetic expression that is then evaluated by a Python interpreter. Gao et al. (2023) show that a stronger approach is one where the model reads the problem and generates the reasoning steps as a program.

Although the cascade reached 88.02% answer accuracy on validation, it scored 7.28% on the test set. Processing single pages rather than the full 8–9 page document recovered at least four points of joint accuracy. The larger cause is block selection: test documents contain decoy blocks that repeat the question's words while asking something else, and 18.9% of the blocks the system selected across the test split match that template. Confabulation by the VLM is present, with about one in four returned identifiers not appearing in the document, but it is too small on its own to account for the gap. This report analyses these failure modes in detail,

along with negative results on standard methods: 3-sample self-consistency (Wang et al., 2023) gave no accuracy gain, and instruction-based reasoning rules stopped working when chain-of-thought (Wei et al., 2022) output is suppressed.

2 Task and Data

The dataset is the one released by Singh et al. (2026). Each instance gives you a PDF, a paraphrased question, and asks for two things back: an answer and an evidence field. The answer is a plain integer. The evidence is a list of the block identifiers the answer came from, as printed on the page. Answers are scored by exact match after normalising, evidence by macro-averaged F1 over the sets, and joint accuracy requires both to be right on the same instance. Joint accuracy is what the leaderboard ranks on.

There are 908 labelled training instances, 217 validation instances whose labels the organisers hold, and 1,730 unlabelled test instances. Every one of the 908 training answers is a plain positive integer, with no units, commas, decimals or minus signs.

The documents have no text layer. Each page is a single JPEG image. The text is drawn, not stored, so pulling it out with a PDF library returns nothing. That leaves two ways in: optical character recognition, or a vision-language model.

The two splits are built differently. Training and validation documents are titled *Regional Operations Brief*, run to two pages, number their blocks b01:, b02: and so on, and hold one solvable problem each. Test documents run 8 or 9 pages, and use different identifiers instead such as R&25:, q-763U:, RAF-D4A:, 74/D4:. They also hold far more blocks.

The splits are also stored at different resolutions. Since every page is one JPEG, you can work out the resolution the corpus actually stores its text at: divide the image’s pixel dimensions by the size it is placed at on the page. Across the first six documents of each split the answer is consistent, and the two ranges do not overlap (Table 4). An identifier set at about 8 pt is roughly 24 pixels tall at 220 dpi and roughly 8 pixels tall at 79 dpi before JPEG compression, and before the diagonal TESTING COPY watermark is laid over it.

3 System

This system is a two-stage cascade built for cost efficiency. Both stages are prompted, not trained, and both run through Anthropic’s API using Claude Sonnet 4.5 and Claude Haiku 4.5. I tried several approaches before settling on this one. The final design uses the cheaper model to locate the right block and the more expensive model to do the reasoning, which took joint accuracy from 82.95% for a single-model version to 88.02%. The system underperformed on the test set, and I identify several failure modes for that below. The stages are described here.

3.1 Rendering

The PDFs have no extractable text layer, so there is no straightforward way to get the text out. I tried both OCR and a VLM. OCR was not effective here: the watermark corrupted Tesseract’s character recognition into garbled strings such as litemns iW in place of "items in the", and secnned relliw in place of "second review". It recovered only 21% of identifiers correctly. Since VLMs handle layout structure as well as text, I went with a VLM. The provider downscales any image wider than 1568 pixels on the long edge, so I chose 135 dpi as a balance between the model’s token budget and keeping small text legible.

3.2 Version 1: Single-Sonnet baseline

The baseline sends all rendered page images of a document to Sonnet 4.5 in a single API call. Sonnet both locates the block and computes the answer, returning the evidence identifier and the arithmetic together. This scored 82.95% answer and joint accuracy on the 217 validation tasks, at an average of \$0.0137 per instance and \$2.97 in total. Evidence F1 was 1.00.

3.3 Version 2: Cascade – Cheap Model Locates, Strong Model Computes

Haiku processes the image tokens to find the target block and transcribe its text, and Sonnet receives text only. Haiku initially returned the identifier with its trailing colon, which is not an exact match against the gold label, so I strip it with `clean_id()`. To keep arithmetic errors out of the model’s hands, Python evaluates the returned expression locally. This gave the best validation accuracy, 88.02% at \$0.0120 per instance. On the test set it scored 7.28%. Section 4 works through why: the largest

factor is block selection, with lower image resolution (71–79 dpi) and confabulation on high-entropy identifiers contributing.

3.4 Version 3: Per-Page Rebuild

Per-page isolation. To reduce confabulation across 8–9 page documents, v3 renders each page separately at 135 dpi and sends one API call per page.

Explicit block selection. The page prompt checks whether a page holds a block beginning `Record <id>`. The file preserves the following case entry.

Tuning the prompt. Every version was tested on the same held-out 20 instances. Tuning meant adding worked examples and rules about how to deliberate. The differences between versions (16/20 for the basic cascade, 18/20 with quantity rules and CoT) come from real changes rather than from fitting the prompt to those instances.

Batch infrastructure. Page images are built in parallel and the resulting requests are submitted through Anthropic’s Batch API.

Refusal fallback. On some test instances the compute stage came back with `stop_reason = refusal` and no content. The request succeeds, so nothing throws an error, it just produces nothing. Since a missing answer counts as wrong, those instances would have been lost. I check `stop_reason` and whether any content came back, and send the same prompt to Haiku 4.5 instead, which answers them.

4 Analysis and Results

Evaluation follows the benchmark’s own definitions: normalised exact match for numeric answers, exact set match for block evidence tokens, and joint accuracy, which requires both to be right on the same instance. System configurations were evaluated on a held-out development set ($n=20$, training instances 200–219, kept apart from the instances 1–5 used to develop the prompts) and on the official validation split ($n=217$).

4.1 Main System Performance & Pipeline Efficiency

Table 1 gives the official scores. On the validation split the cascade (V2) reached 88.02% answer accuracy with an evidence F1 of 1.00, improving on

Split	System	Answer	F1	Joint
Validation	V1: single model	82.95	1.00	82.95
Validation	V2: cascade	88.02	1.00	88.02
Validation	V2 + refinements	87.56	1.00	87.56
Test	V2: same system	7.28	10.17	—
Test	V3: per-page	—	—	11.21

Table 1: Official scores (%). System V2 is the only system evaluated on both splits.

Configuration	Answer	Evidence	\$/inst.
Single model, vision + arith.	17/20	—	0.0140
Cascade	16/20	20/20	0.0082
+ CoT + quantity rules	18/20	20/20	0.0120
+ 3-sample self-consist.	18/20	20/20	0.0282

Table 2: Configurations on held-out training instances 200–219 ($n=20$).

the single-model baseline (V1) by just over five points. On the held-out test split the same system, unchanged in every respect, obtained 7.28% answer accuracy. A revised system (V3), which issues one request per page rather than one per document, scored 11.21% joint accuracy.

The 88.02-to-7.28 comparison is the only controlled one available to me: identical code, identical prompts, identical models, two splits. It is an 81-point collapse, and the remainder of this paper is an account of it.

Table 2 reports the configurations compared during development, on training instances 200–219 held out from prompt construction. Four observations follow from it, and one goes against conventional thinking.

First, the cascade architecture is substantially cheaper. Moving the stronger model off the page images and onto approximately 300 tokens of extracted text reduces cost from \$0.0140 to \$0.0082 per instance, a 41% reduction, at 16/20 against the baseline’s 17/20. At $n = 20$ and accuracy near 0.9 the standard error of a proportion is roughly ± 13 points.

Second, the configuration actually submitted costs \$0.0120, a 14% reduction rather than 41%, and scores 18/20 on development data and 88.02% on validation, against the baseline’s 17/20 and 82.95%. That configuration dominates the single-model baseline on both accuracy and cost, which is the claim made. The 41% figure belongs to a configuration I did not submit, and I separate the two because the distinction is easy to lose.

Third, the two prompt interventions do nothing in isolation and something together. Chain-of-thought alone leaves accuracy at 16/20. The

Configuration	\$/inst.	1,730 inst.
Single model	0.0140	\$64
Cascade	0.0082	\$29
Cascade, Batch API	0.0041	\$15

Table 3: Cost. Rates are \$1/\$5 per million input/output tokens for Haiku 4.5 and \$3/\$15 for Sonnet 4.5.

instruction to use every stated quantity, applied without visible reasoning, also leaves it at 16/20. Applied together, at identical prompt text, accuracy moves to 18/20.

Finally, self-consistency is a clear negative result in this task. Three independent samples per instance agreed on all 20 instances tested, so the majority vote is identical to a single sample while costing \$0.0282, which is twice the single-model baseline for no measured change in accuracy.

The rates in Tables 2 and 3 are measured on the two-page development and validation documents. The single-model rate held at \$0.0137 over the full 217-instance validation split, a run that cost \$2.97, against \$0.0140 on the twenty development instances. Test documents run to eight or nine pages, so the locate stage costs more there while the text-only compute stage does not change; the 1,730-instance totals are projections on that basis.

4.2 What Resolution Buys

The two splits are not stored the same way. Every page is a single JPEG, so the resolution the corpus holds its text at can be computed directly: divide the image’s pixel size by the size it is placed at on the page. Across the first six documents of each split the answer is consistent and the ranges do not overlap (Table 4). Training pages are 220 dpi. Test pages are 71–79 dpi. An identifier set at about 8 pt is roughly 24 pixels tall in the first case and roughly 8 in the second, before JPEG compression and before the watermark is laid over it. This makes identifiers harder to read, but it does not explain the collapse. Table 5 shows what happens when I ask a model to transcribe every identifier on three pages of a test document and score the result against 48 identifiers I read by hand. At 135 dpi, Haiku recovers 36 of 48 and Sonnet recovers 39. At the source’s own 79 dpi, Sonnet recovers 11. Identifiers are the hardest thing on the page because there is nothing to guess from. A blurry word in a sentence can be worked out from the words around it. R09–A7F cannot, because every character stands alone. When the model cannot read the string it returns something shaped like an identifier that is not

Split	Pages	Embedded JPEG	Native
Train (6 of 6)	2	1786 × 2526	220 dpi
Val (6 of 6)	2	1310 × 1853	~162 dpi
Test (4 of 6)	8–9	572 × 809	71 dpi
Test (2 of 6)	8	643 × 910	79 dpi

Table 4: Resolution of the single JPEG comprising each page, over the first six documents of each split. An 8 pt identifier is roughly 24 px tall at 220 dpi and roughly 8 px at 79 dpi.

Model	Render	Exact	Not found
Haiku 4.5	135 dpi	36/48 (75%)	12
Sonnet 4.5	135 dpi	39/48 (81%)	9
Sonnet 4.5	79 dpi	11/48 (23%)	27

Table 5: Identifier transcription against 48 hand-verified identifiers on three pages of task_010001. The third row renders at the source’s native resolution.

in the document, because filling in a likely answer is what it does once the evidence runs out.

Enlarging the image matters far more than the model does. The two Sonnet rows are the same model on the same JPEG, 58 points apart, against six points for switching model at the same resolution. The likely reason is how the encoder splits an image into fixed patches: at 79 dpi an 8-pixel glyph sits inside one patch and is averaged into a single token, while the enlarged version spans two or three. Enlarging adds no detail, but it adds tokens, and for small text the tokens are what is short.

Resolution does not degrade performance smoothly. The drop from 220 to roughly 162 dpi between the development and validation splits cost nothing measurable, while the further drop into the test range coincided with an 81-point fall. Legibility appears to fail at a threshold rather than decline in proportion.

Accuracy also depends on what is being read. Table 6 measures three kinds of content on the same pages at four render resolutions, and the ordering holds at every one: word recall reaches 99.5% by

Render	Prose	Numbers	Identifiers
79 dpi (native)	80	59	19
100 dpi	98	87	62
135 dpi	100	96	77
200 dpi	100	98	71

Table 6: Recovery by content type and resolution (%), Haiku 4.5 on eight blocks from pages 1–3 of task_010001. Prose is unordered word recall; numbers and identifiers are exact match, so the columns are not directly comparable.

135 dpi, numbers 96%, and identifiers 77%, which is the highest they reach anywhere in the range tested. The difference is redundancy. A misread word is constrained by the words around it; a misread digit only by the arithmetic; an identifier such as R24-82F by nothing at all. Resolution is worth most where redundancy is least, and exact match over a high-entropy string admits no partial credit.

4.3 Classical OCR Baseline Evaluation and Rejection

Classical OCR (Tesseract) was evaluated and rejected. Against the 48 hand-verified identifiers it recovered 10 exactly (21%), which is disqualifying under exact-match evidence scoring; throughput was 13.67 s per page across eight threads, projecting to roughly 52 hours for the test split.

Its failures are, however, of a different kind. Consider task_011327, the instance where the vision pipeline returned an invented identifier, record number, and both quantities. OCR returned:

```
Grace counted (13+ 17)litenms
iW'the first review anct 13 in
the secnned relliw.
```

The prose is corrupt; the quantities are exact, and they are the ones the gold answer requires. `litenms iW'the` is self-evidently a failure to read. R09-A7F is indistinguishable from a genuine identifier without the document in hand. Two methods failed on the same page and only the weaker one signalled it.

5 Conclusion

Moving the strong model off the page images and onto extracted text made the system both cheaper and more accurate on validation: 88.02% against 82.95%, at \$0.0120 against \$0.0140 per instance. The same system, unchanged, scored 7.28% on the test split, and two things explain it. The first is selection: test documents contain decoy blocks that reproduce the question's words while asking something else. The second is resolution. The splits are rendered differently. Training pages hold their text at 220 dpi, validation at 162, and test at 71 to 79. The step from 220 to 162 cost nothing measurable. The step below it coincided with an 81-point fall, which suggests legibility fails at a threshold rather than declining in proportion. Resolution at inference time matters more than model choice. Rendering the same JPEG larger took identifier recovery

from 23% to 81%, while moving from Haiku to Sonnet at the same resolution took it from 75% to 81%. Enlarging adds no detail, but it gives small glyphs more tokens to occupy. Since the resolution cap applies to whatever image is sent, sending page regions rather than whole pages might give each block more of that budget.

Limitations

Identifier transcription and block selection both contribute to the collapse, and this paper does not measure their relative weight. Because the content words overlap between the gold block and the decoys, the ranking ties, and the choice falls to whichever candidate happens to appear first in the results list. Scoring candidates on what a block asks rather than which words it contains separates the target from all three decoys in the document I enumerated; I could not evaluate this at scale, since test labels were not released. The block filter used on the test split was derived from the released test documents rather than from training data. Test scores were withheld after the first attempt, so the diagnosis in Section 4 proceeded without gold labels. The decomposition of the collapse into identifier reading and block selection is consistent with the observed score but is not measured against it.

Acknowledgments

I thank the organizers for putting this shared task together. The system, experiments, and measurements reported in this paper are my own. An AI assistant was used solely to improve the language of the original draft, which I reviewed and verified against my own experiment logs.

References

- Lingjiao Chen, Matei Zaharia, and James Zou. 2023. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*.
- Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. 2021. DocVQA: A dataset for VQA on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2200–2209.

- Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. 2023. Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–840.
- Jyotika Singh, Fang Tu, Aziza Mirsaidova, Amit Agarwal, Hitesh Laxmichand Patel, Sandip Ghoshal, Miguel Ballesteros, Karan Dua, Yassine Benajiba, Weiyi Sun, Tao Sheng, Graham Horwood, Sujith Ravi, and Dan Roth. 2026. [Gsm-sem: Benchmark and framework for generating semantically variant augmentations](#). *arXiv preprint*.
- Ray Smith. 2007. An overview of the Tesseract OCR engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR)*.
- Xuezhi Wang et al. 2023. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*.
- Jason Wei et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*.