Most OCR comparisons still start with the benchmark table.

The harder production question is simpler: which model breaks least often on the pages you actually have.

This guide is organised around that question. On our scan-heavy OCR pilot, the useful conclusion was not one universal winner. It was a routing rule:

- `FireRed-OCR` became the best default for text-first pages once its wrapper handled blank pages and preserved page images
- `GLM-OCR` stayed safer when the question depends on a small inline graph, apparatus, particle diagram, or reaction scheme
- `dots.ocr-1.5` was more compelling when OCR was only one part of a broader visual parsing workflow
- `PaddleOCR-VL-1.5` stayed relevant when a team wanted a mature OCR baseline tied to a broader parsing ecosystem

For the full scan-heavy benchmark method and the evidence behind this routing rule, see: https://instavar.com/blog/ai-production-stack/How_We_Benchmark_OCR_Models_on_Scan_Heavy_PDFs.

For the wider market map, see: https://instavar.com/blog/ai-production-stack/OCR_SOTA_Feb_2026_Open_Document_AI_Leaderboard.

Start here

If you only need the first pass:

- choose `FireRed-OCR` when the page is mostly notes, bullets, tables, worked answers, or formulas
- choose `GLM-OCR` when the page is really asking the reader to interpret a small local visual
- choose `dots.ocr-1.5` when OCR is only one piece of a larger visual-language parsing workflow
- keep `PaddleOCR-VL-1.5` in the shortlist if you want a mature baseline with a broader surrounding document stack

1 The shortest workflow-fit answer

Your workflow bottleneck	Best first model to test	Why
Cleanup cost on text-heavy scans	`FireRed-OCR`	It became the cleanest Markdown-first default on text-first pages in the patched pilot
Diagram-linked question pages	`GLM-OCR`	It preserved question-local visuals more safely than the other stacks
OCR plus web, screen, scene, or SVG-style parsing	`dots.ocr-1.5`	It is the clearest broader parser in this group, not only a document OCR model
Mature baseline plus ecosystem depth	`PaddleOCR-VL-1.5`	It remains a strong OCR baseline with a wider surrounding parsing ecosystem
Mixed PDFs that alternate between notes and figure-heavy worksheet pages	`Hybrid routing`	Different page types favored different models in practice

2 What the pilot changed

The scan-heavy pilot did not support a simple “best OCR model in 2026” claim.

What changed:

- early raw comparisons made `GLM-OCR` look like the safest overall default
- once the `FireRed-OCR` wrapper stopped hallucinating on near-blank pages and preserved page images, the result shifted materially
- after that patch, `FireRed-OCR` led `24/31` documents in the final `5`-way run, but the remaining `GLM-OCR` wins were still meaningful because they were diagram-question-heavy

The practical result was that model choice changed with page type.

3 When to use FireRed-OCR

Use `FireRed-OCR` first when the expensive failure mode is structural cleanup on text-first pages.

Good fit:

- revision notes
- bullet-heavy teaching pages
- answer keys
- formula-heavy explanations
- tables that still need to read linearly in Markdown

Why it works:

- the structural OCR focus is real, not just branding
- once blank-page handling and page-image preservation were fixed, its text-first output was often cleaner than the compact OCR baselines
- it is the better default when the page is mostly meant to be read top-to-bottom

Watch-outs:

- pages where the answer depends on a small local visual instead of the surrounding prose
- worksheet pages with many inline answer-option figures
- pages where a reaction scheme or apparatus needs to stay tied to a specific nearby sentence

`FireRed-OCR` is the best first test when the main cost is messy Markdown, not missing diagrams.

4 When to use GLM-OCR

Use `GLM-OCR` first when the page is visually local and the text depends on that locality.

Good fit:

- question pages that say “the diagram below”
- apparatus-linked practical questions
- particle-box choices
- reaction-network questions
- small graphs embedded inside a worksheet question

Why it works:

- it preserved inline regions more safely than the other models in the pilot
- it remained the safer choice on diagram-question-heavy documents even after the `FireRed-OCR` pipeline improved

- the production issue here is not only recognition accuracy but keeping the right visual tied to the right question

Watch-outs:

- long text-heavy notes where its raw Markdown tends to be noisier than the best `FireRed-OCR` output
- pages where the main job is reading prose, tables, or answers, not preserving local diagrams

If the question breaks once the inline figure disappears, `GLM-OCR` should stay in the routing path.

5 When to use dots.ocr-1.5

Use `dots.ocr-1.5` when your real requirement is broader than document OCR.

Good fit:

- OCR plus web page parsing
- OCR plus screen parsing
- OCR plus scene text
- workflows where SVG-like structure or non-document visual parsing also matters

Why it works:

- it is positioned more clearly as a broader visual parser than as a narrow document OCR specialist
- it deserves a slot when one stack may need to cover several parsing modes, not just scanned PDFs

Watch-outs:

- as the default OCR engine for scan-heavy school notes or worksheets
- when your hardest pages are text-heavy PDFs and your main cost is Markdown cleanup

In the internal pilot, `dots.ocr-1.5` won only `2/31` documents in both the raw `3`-way and patched `5`-way comparisons. That does not make it unimportant. It means its main value is different from the main value of `GLM-OCR` or `FireRed-OCR`.

6 When to keep PaddleOCR-VL-1.5 in the shortlist

Keep `PaddleOCR-VL-1.5` in the shortlist when you want a strong OCR baseline with a broader ecosystem around it.

Good fit:

- teams that care about ecosystem maturity as much as one model score
- document pipelines that may expand into broader modular parsing workflows
- OCR teams that want a serious baseline even if it is not the final default for every page type

Why it still belongs:

- it is still one of the strongest public OCR baselines in this generation
- it has a fuller surrounding ecosystem than the newer challengers
- it is useful when the model decision is really a stack decision

Watch-outs:

- in this specific Markdown-first, scan-heavy comparison, it was not the main story once `FireRed-OCR` was patched and `GLM-OCR` was already present as the diagram-safe baseline

7 The practical routing rule

If you only want one routing policy from this article, use this:

1. If the page is mostly notes, bullets, tables, answers, or formulas, start with `FireRed-OCR`.
2. If the page depends on a small inline graph, apparatus, particle diagram, or reaction scheme, route it to `GLM-OCR`.
3. If the workflow needs OCR plus broader web, screen, scene, or SVG-style parsing, evaluate `dots.ocr-1.5` as a separate lane.
4. Keep `PaddleOCR-VL-1.5` as a mature baseline when the surrounding parsing ecosystem matters, not just one page-level outcome.

That routing rule is more useful in production than arguing about a single universal winner.

8 What this means for mixed documents

Some PDFs should not be assigned to one model end to end.

That was the clearest lesson from the mixed chemistry worksheet packs in the pilot:

- text-heavy note pages often favored `FireRed-OCR`
- diagram-question pages often favored `GLM-OCR`
- broader visual parsing questions still justified a separate `dots.ocr-1.5` lane

If your corpus mixes all three, treat routing as part of the product. Do not wait to invent routing after rollout.

9 Bottom line

The right OCR choice in 2026 depends less on one benchmark headline than on the kind of page that hurts you when it breaks.

- choose `FireRed-OCR` when cleanup cost on text-first pages is the bottleneck
- choose `GLM-OCR` when question-local visuals have to stay attached to the right text
- choose `dots.ocr-1.5` when OCR is only one part of a broader parser
- keep `PaddleOCR-VL-1.5` in the shortlist when ecosystem depth matters

That is the cleaner way to choose a stack than publishing another “best OCR model” table without page semantics.

Sources

- How We Benchmark OCR Models on Scan-Heavy PDFs
- OCR SOTA Feb 2026 Open Document AI Leaderboard and Deployment Guide
- GLM-OCR repository
- GLM-OCR model card
- dots.ocr repository
- dots.ocr paper (base model)
- PaddleOCR-VL-1.5 paper
- PaddleOCR-VL-1.5 model card
- FireRed-OCR repository
- FireRed-OCR model card
- FireRed-OCR paper