THE DOCUMENT AI ROADMAP

Seven steps to better results, lower cost, and fewer surprises.

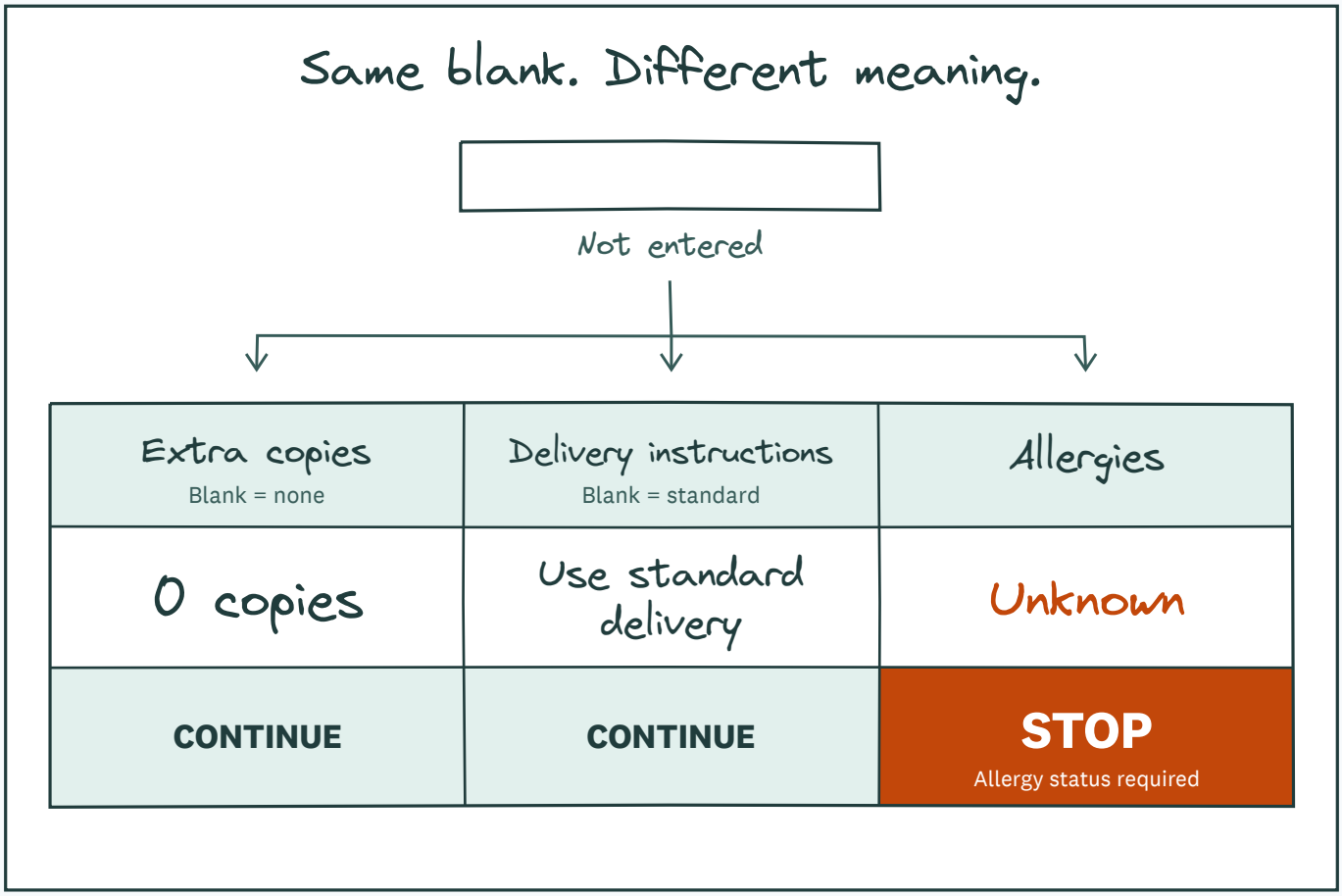
A practical guide by Isaac Flath

For people building products and workflows with documents.

Choosing a model is step 1. Here's the whole solution.

Most people stop here	01	Model Selection
Quality	02	Data Annotation
	03	Error Analysis
Reliability	04	Evals
	05	Observability and Tracing
Cost	06	Routing and Pipelines
	07	Caching and Batching

Misconception 1:
The benchmark winner is best for me



"Correct" depends on what your product and user need the output for. How a blank value should be handled, and how critical it is cannot be generally handled without the individual product context.

A benchmark might punish recording blanks as 0 as a failure when it doesn't matter for your use case at all. Or it might not punish or test for it much when it is an absolutely critical legal requirement.

The benchmark score does not tell you any of this and the more of these misalignments there are, the less meaningful the benchmark score is.

No benchmark can do this "right". Because there is no universal "right", it's project-based.

Misconception 2: Benchmark PDFs are like mine

Both: "complex tables"

BENCHMARK COULD TEST

Region	FY 2025	
	Actual	Plan
North	1,280*	1,300
	(35)	-
South	970 ~~960~~	1,050

* less returns

YOUR PDF MIGHT HAVE

Metric	FY 2025	
	Q1	Q2
Revenue ($m)	12.4 / 12.0	13.1 / 14.0
Margin (%)	18 / 20	22 / 20
Refunds ($k)	48 / 50	44 / 50

Actual / plan; color = status

On target Off target

A benchmark cannot contain every kind of PDF. "Complex tables" is unusually clear but still could mean color-coded forecasts, actual / plan values in one cell, merged headers, wide tables, handwritten adjustments or about 10,000 other variations. Sometimes actual benchmark pages and scoring rules match your product exactly, but not always.

It's like asking where an event is, and a friend says it's in the Midwestern United States. That's a useful start because you get a good idea on what the flight time and travel cost will be. But if you're trying to book a hotel room and flight right now it's totally useless.

The same applies to building document workflows. Benchmarks are great general signals but don't give all the information you need to create a reliable product or workflow.

Even vendors with their own benchmarks ask to evaluate your specific PDFs when there's a large enterprise offering, not only rely on their benchmarks. While that extra evaluation is useful, it's a search across that particular vendor's offering not a search for the best offering for the use case regardless of the source.

STEP 1 OF 7 / MOST PEOPLE STOP HERE

01 Choose a model

Example: a lender reviewing annual reports

The product needs...	~~Model A~~	Model B	~~Model C~~	Model D
Get revenue as structured data	✓	✓	✓	✓
Click a number to check the source	✗	✓	✓	✓
Keep files inside our environment	✓	✓	✗	✓

Shortlist: B D Pick one. Start with B.

Start by eliminating model choices based on product requirements.

Markdown is great for feeding text to an LLM, but it cannot handle complex tables with merged cells. For that, you need HTML. But if your goal is to extract data from the documents to store in a database, you need a different approach that returns JSON.

Then, consider what action the user needs to take. A lawyer does research on prior cases so they can file an argument with the court. Or a doctor wants a timeline of a patient's medical history. They CANNOT complete their work without citations to source records with bounding boxes, so the output format must support that.

Finally, there are compliance requirements, like on-premise or regional hosting that eliminate some models or vendors instantly.

Find the models that meet your structural needs first, before testing on your documents.

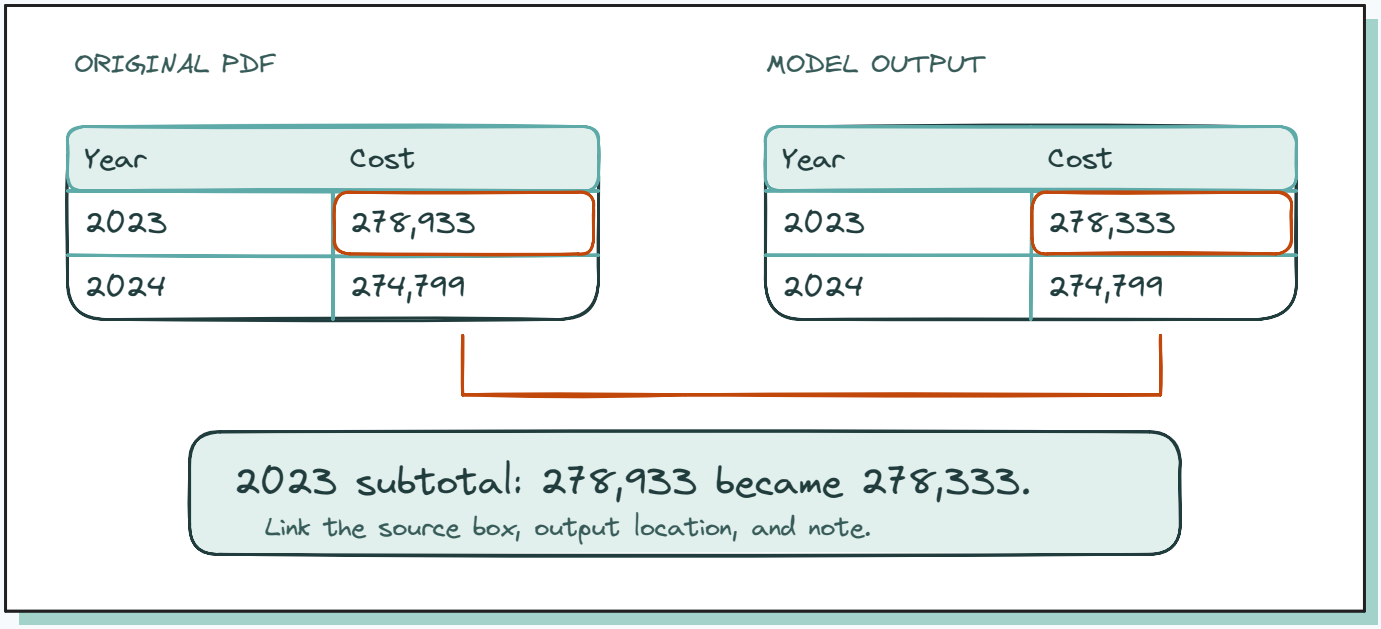
READ MORE

How to Choose an OCR Model

VLM OCR for Hard Documents

STEP 2 OF 7 / QUALITY

02 Annotate the mistakes



Compare OCR model outputs with the original PDF to find errors. An error is a difference that would hurt the user downstream, not all differences.

In the table above, OCR misread 278,933 as 278,333. To annotate this, write “wrong value” and attach that to a bounding box on the original PDF and the location in the model output.

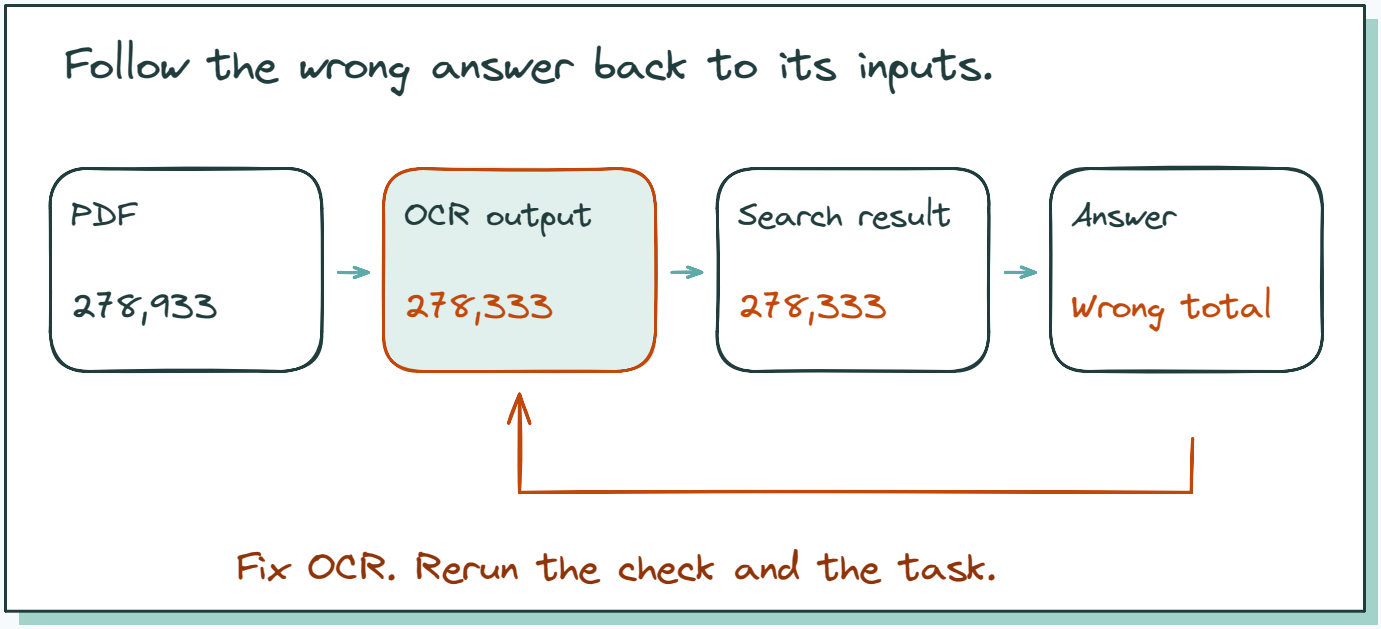
Keeping those three pieces together lets you retest the error with a new pipeline without repeating the review. You have the original value, what the model produced, and a clear description of what needs fixing.

To do this efficiently you need to build a custom annotation app. This used to be done via hand-coded widgets in notebooks, but now coding agents allow you to make much better interfaces with much less effort.

READ MORE

Document Evals: OCR, Retrieval, and Agents

03 Trace and fix the failures



When an agent answers a question incorrectly, you need to know where the error is to fix it. The error may have happened before the agent received it.

For example, a calculation could use 278,333 (which is wrong). If we look at a log of the agent history we can see that wrong number in the context window, so we know it's not an agent hallucination.

So we need to trace it back to see how it got there. In this case, from a retrieval tool call. We continue tracing back to the source to find retrieval found the correct document, but the OCR model changed 278,933 to 278,333.

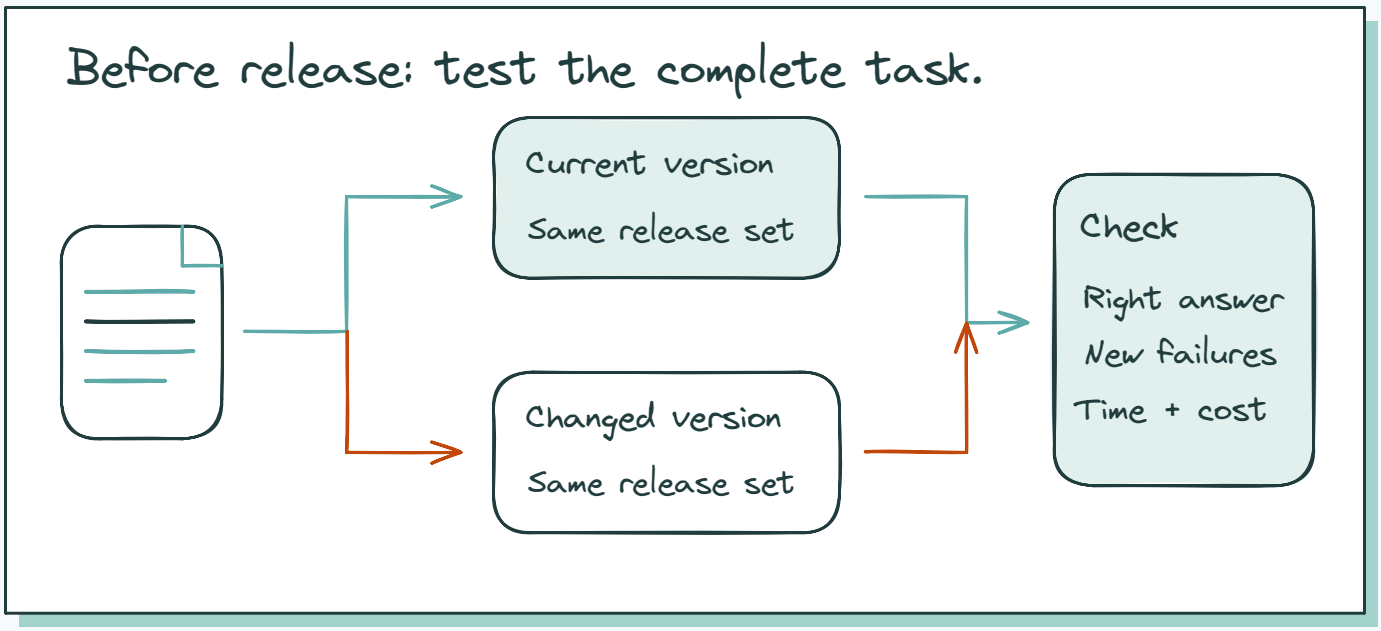
Following the data backward locates the failure in extraction. If the problem was document extraction no prompt tuning or agent fix will improve your system.

You have now isolated the root cause of the quality issue and can fix it.

READ MORE

Formative vs. Summative Evals for Document AI

04 Test changes before shipping



Now that the quality issue has been traced, we need to make tests. This lets us iterate and improve the system, and gives confidence that future changes will not cause this error to come back.

There are two kinds of evals, summative and formative.

Summative evals measure whether the answer did what the user needed. Were the assembly instructions good enough to let the customer construct their furniture? Did the answer and citation let the user write their report quickly and correctly? But these are expensive and only tell you overall health, not how to fix errors.

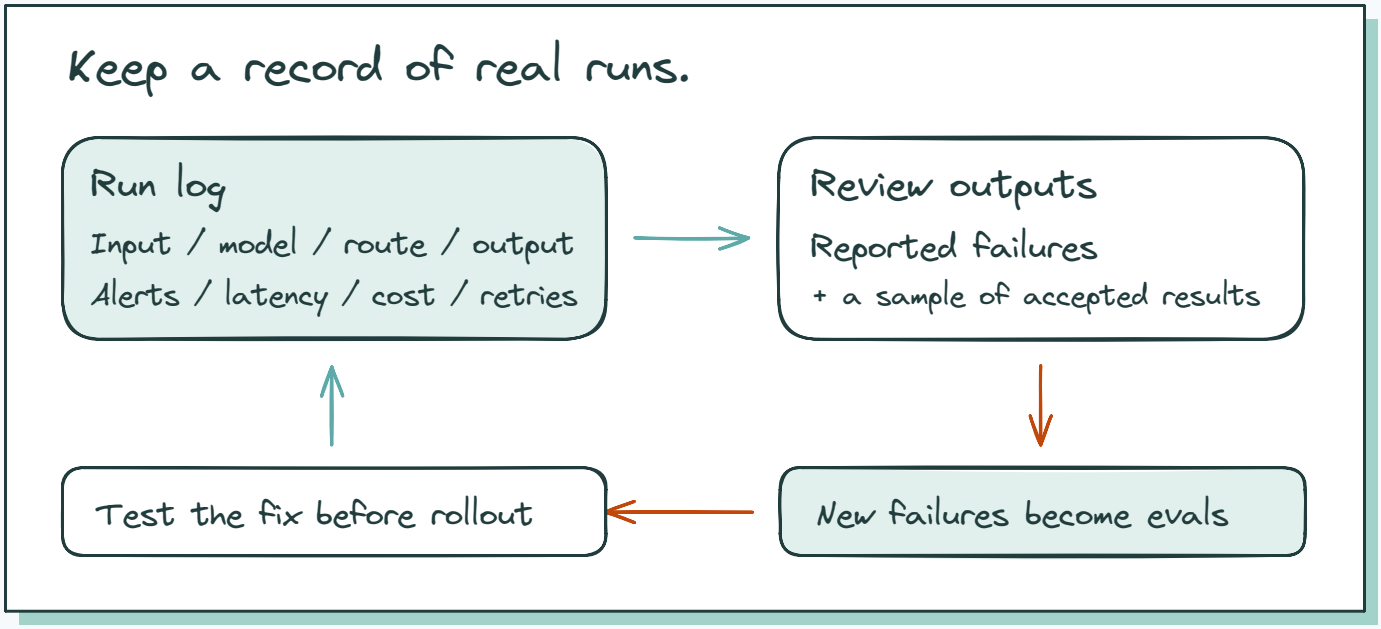
Formative evals target problem areas of the system so you can iterate and improve. Did the OCR model convert the table to HTML accurately? Did retrieval find the right document for the user question?

You need both to build a reliable system.

READ MORE

Formative vs. Summative Evals for Document AI

05 Monitor real runs and catch new failures



An agent may search, crop pages, and run OCR several times while answering one question. Every step needs a record of all of this so you can debug and improve the system.

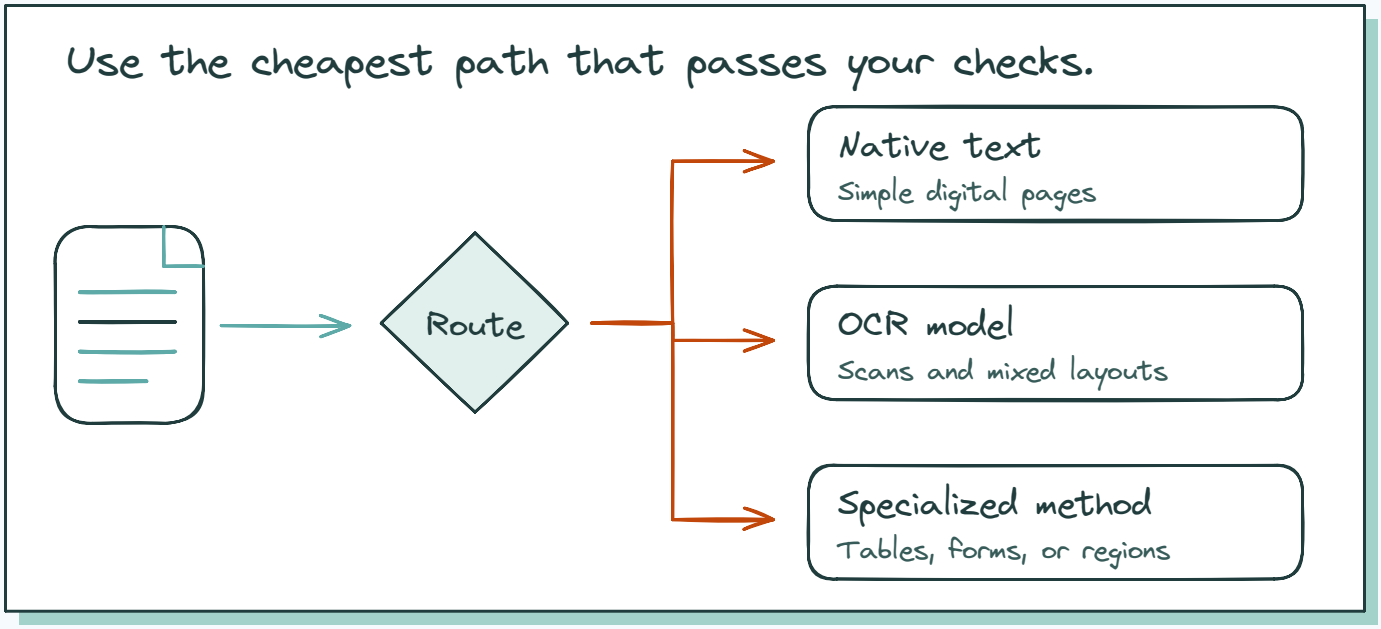
For example, an agent can spend several minutes looking for an answer that isn't in the documents. The final response won't show whether it searched once, or repeatedly cropped and OCRed the same pages in an expensive loop.

Being able to diagnose and understand what went wrong and where is the most important part of an AI application because it tells you if your system is working properly.

Because of that, I prefer to use standard application storage (like Postgres) to have the most flexibility in this process.

Reviewing these records turns the checks from the earlier steps into a long-term repeatable and automatable process.

06 Route to the cheapest method that works



More accurate models are slower and more expensive. Routing is the first step to a pipeline that gives you high accuracy for low cost. Almost all OCR vendors do this for their services to increase margin.

For example, a three-page report may include a title page, a recommendation page, and a supporting data page. The title page can almost certainly be parsed with an extremely cheap model. No need for a heavy VLM.

The recommendation page could be parsed mostly with a fast pipeline, which then crops just one supporting figure to send to Qwen or another VLM. And the data page may need to be sent in full to a VLM to handle the complex table.

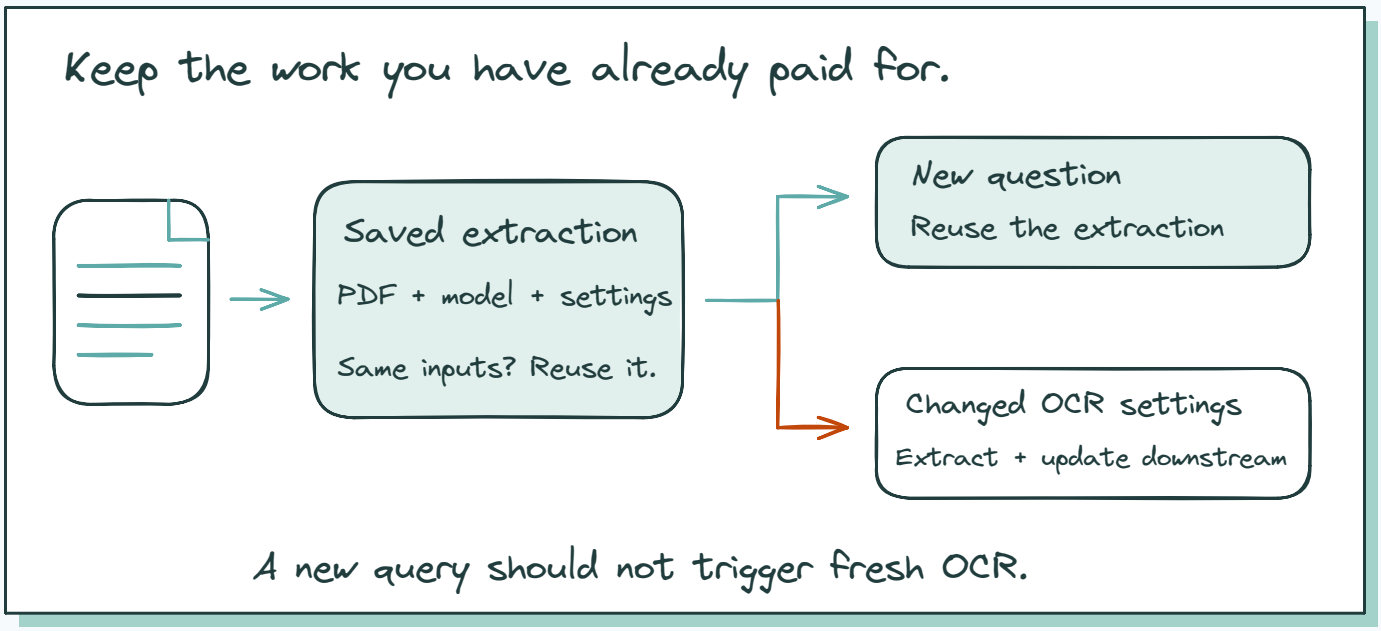
Or leasing paperwork may be 30 pages of standard typed text, then 2 pages of handwritten forms and signatures. Handwriting and signatures require a more expensive model than pages of typed text.

A router lets you define logic to do this. It can be rule-based, then level up to a small trained regression model for more accurate routing.

READ MORE

OCR Routers: When Should You Use a Bigger Model?

07 Stop paying for repeated work



In many cases up to 30% of pages submitted to OCR are duplicates that can be looked up instead of processed again.

Examples include pages from standard templates, or just documents the same users upload. For example, purchasers in the same company may review the same information and invoices at different times.

By saving the PDF and outputs, you can look up if you already have processed that document. If so, return that output for the page instead of doing OCR again. This often leads to significant cost savings, especially if paying a vendor per page.

You do need to be careful to update caches when you improve your pipeline though, otherwise you serve old lower quality results forever.

A second cost reduction can be to batch at the cost of latency. Many services provide a 24-hour turnaround on OCR, which lets them run huge batches overnight.

Batching often works well for products where their primary use case is analytical, such as processing monthly financial statements, or daily medical records where next-day availability is standard.

Build your document workflow

This guide is a quick overview of the key components we'll cover in PDF to Production.

Joe Barrow and I will show how to build a custom document-processing pipeline with vision-language models (VLMs), using realistic PDFs. We'll compare outputs, investigate mistakes, and build custom evaluations for the tasks your product needs to do.

We'll also cover deployment and leave time for questions about your documents and workflow.

Explore the course

isaacflath.com/pdf2production