From OCR to Visual RAG

A PRACTICAL GUIDE TO MODERN DOCUMENT UNDERSTANDING WITH PALI, COLPALI, AND DOCVQA

Markus Kuehnle

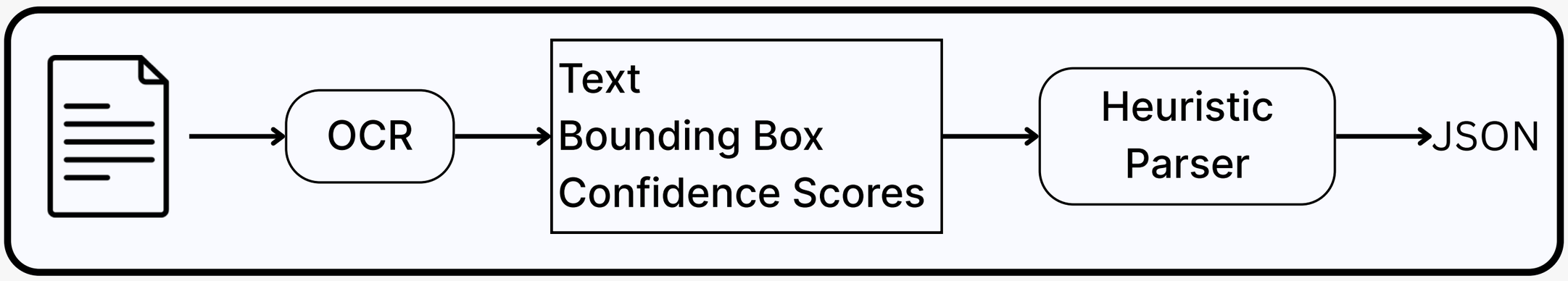
Document understanding is one of the most common AI problems

- Forms, contracts, invoices, and reports are central to business workflows.
- Classic OCR pipelines extract text but fail to understand layout, semantics, or visual structure.
- Vision-Language (VL) models aim to bridge this gap by learning from both text and image jointly.

This guide breaks down modern approaches:
→ When OCR still matters
→ Where VL models outperform
→ How tools like PaLI and CoPaLI fit into real-world RAG pipelines

1 How Traditional OCR Pipelines Work

The Classic Stack: OCR + Heuristics

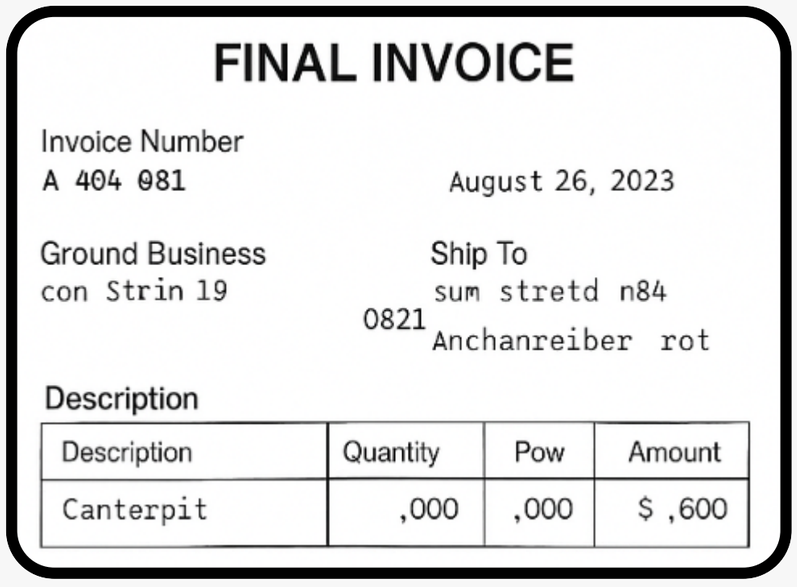


- Convert scanned document → text via OCR
- Parse layout manually: tables, headers, sections
- Extract fields with regex, keywords, or position logic
- Patch edge cases with rules

2 The Problem with Traditional OCR

Common Issues

Layout	Languages
Orientation	Handwriting
Tables	Artifacts
Columns	Contrast



FINAL INVOICE

Invoice Number
A 404 081

August 26, 2023

Ground Business
con Strin 19

Ship To
sum stretd n84
0821 Anchanreiber rot

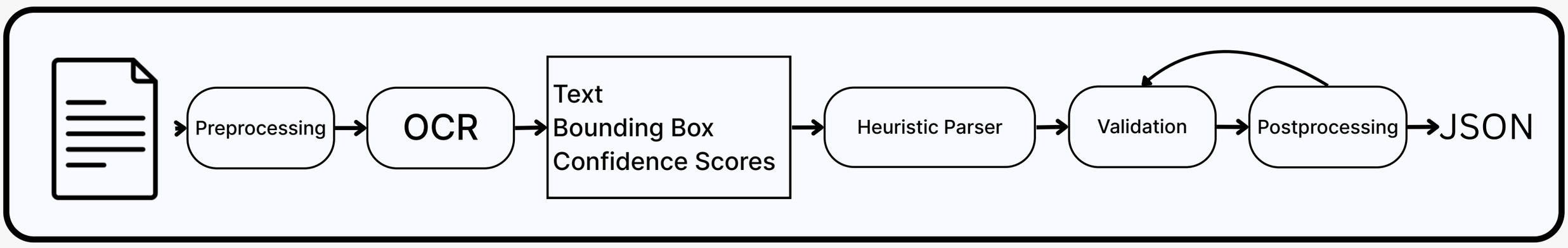
Description

Description	Quantity	Pow	Amount
Canterpit	,000	,000	$,600

Solutions: Pre- and Postprocessing

- Rotation / Deskewing
- Histogram Equalization
- Binarization
- Denoising
- Morphological Ops
- Resolution Upscaling
- Bounding Box Filtering
- Language-specific Models
- Layout Detection

... but suddenly your pipeline looks like:



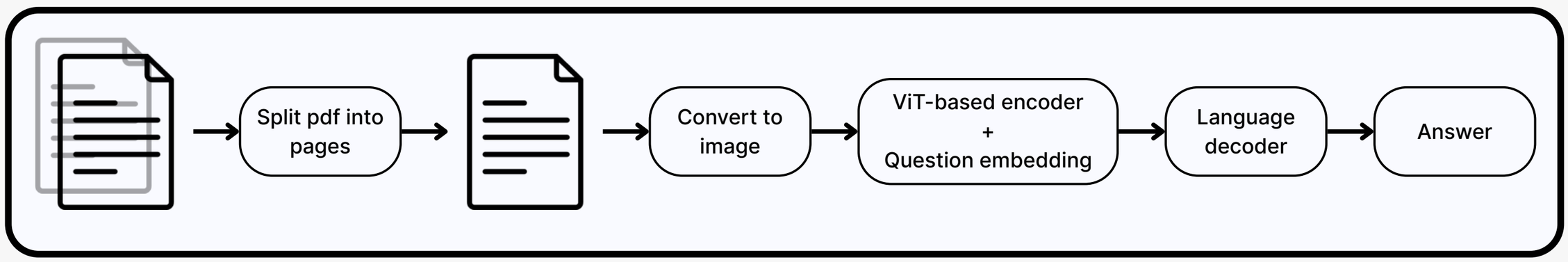
3 A Smarter Approach: Visual Question Answering (VQA)

Instead of parsing raw text, we ask the model questions about the document image:

- "What is the invoice total?"
- "Who is the sender?"
- "What date is mentioned?"

This is called **Document Visual Question Answering (DocVQA)**.

- It learns structure visually (no need for bounding boxes)
- Works across document types and layouts
- And no parsing logic required



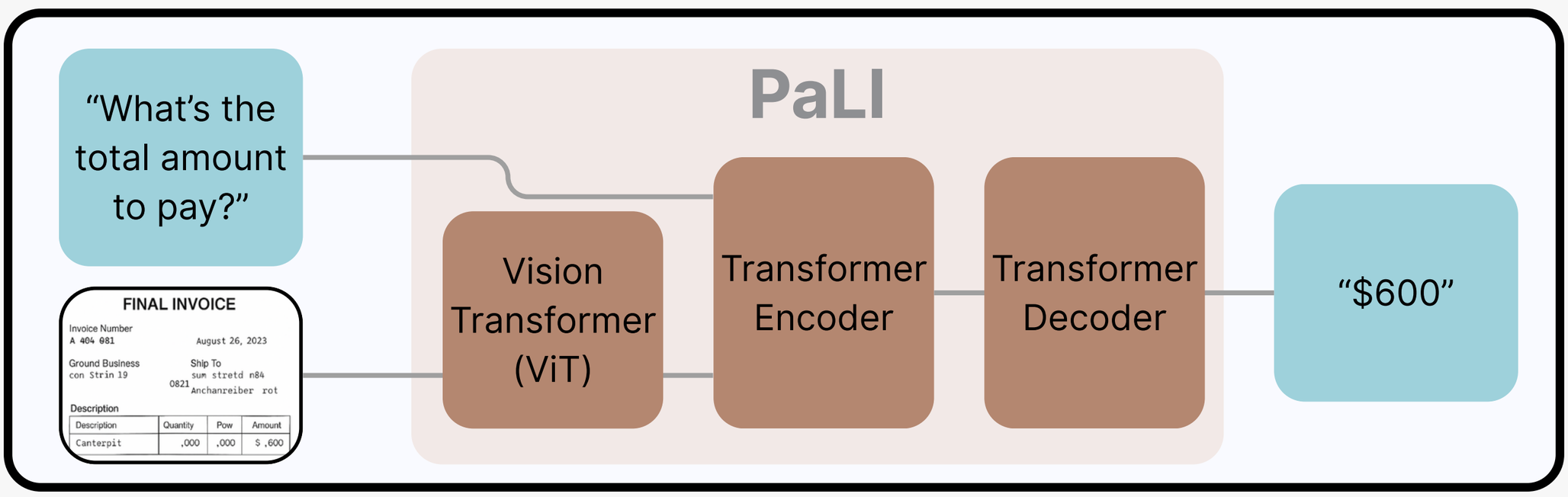
Most RAG systems break on scanned docs.
PaLI + ColPaLI let you query PDFs natively.. no OCR pre-parsing.

4 PaLI: Multilingual, Multimodal, Scaled VQA

PaLI (Pathways Language and Image model) is Google's massive Vision+Language model

It handles:

- OCR-free VQA
- Image captioning
- Multilingual understanding
- Zero-shot generalization



- It works across scanned documents, charts, forms, handwritten notes
- Answers open-ended questions without needing task-specific tuning
- Massive scale = robust performance in noisy, multilingual, layout-heavy settings

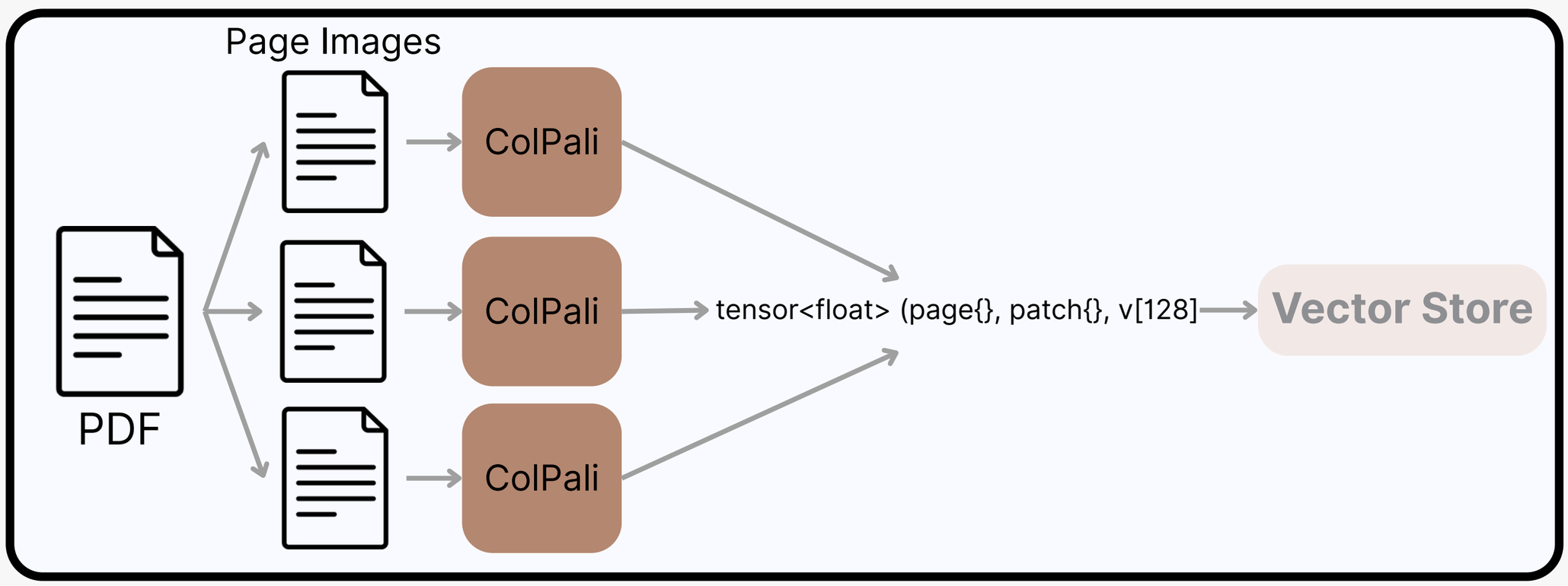
5 ColPaLI: Visual RAG for Document Retrieval

ColPaLI = PaLI-Gemma encoder + ColBERT-style retriever

What it does:

- Retrieves relevant document pages before answering
- Enables RAG pipelines for layout-rich, scanned documents
- Uses token-wise embeddings (not single vectors)
- Preserves positional + visual structure during retrieval
- Ideal for fine-grained similarity (tables, forms, spatial layout)

Instead of treating each doc as one vector, ColPaLI compares tokens, making it perfect for documents where layout and structure matter.



6 Visual RAG: Combining Retrieval + VQA Without OCR

Problem:
Traditional RAG relies on OCR text for retrieval.
But scanned PDFs often lack high-quality text.

Solution: Image-based retrieval with ColPaLI

- ColPaLI embeds full document images into dense vectors
- Store these embeddings in a vector DB (e.g. Vespa, FAISS)
- Embed the user query and retrieve the most similar pages
- No OCR needed, retrieval works on visual similarity

Extract answers using VQA (Visual Question Answering)

- Feed the retrieved image + query into a VQA model (e.g. PaLI, Pix2Struct)
- Get precise, context-aware answers from layout-rich pages
- Useful for invoices, receipts, forms, handwriting, etc.

Final Step: Use VQA output in your RAG system

- Replace traditional retrieved text chunks with VQA answers
- Or combine both (if OCR is available) for hybrid retrieval

Example:
From Similar Pages to Answers

Step 1: Use ColPaLI to embed scanned documents
→ enables dense retrieval over full-page visual context

Step 2: For top-k matches, use a VQA model (e.g., PaLI)
→ answer questions directly from the image

Outcome: No OCR. No heuristic parsing. Just questions and answers.

```python
from transformers import ColPaliProcessor, ColPaliForRetrieval, PaLIProcessor, PaLIFinQA
from PIL import Image
import torch

# Load ColPaLI model and processor
retriever = ColPaliForRetrieval.from_pretrained("vidore/colpali-v1.3-hf", device_map="auto")
retriever_proc = ColPaliProcessor.from_pretrained("vidore/colpali-v1.3-hf")

# Embed a scanned document
image = Image.open("invoice_293.png")
query = "What's the total amount on invoice #293?"

inputs = retriever_proc(text=query, images=image, return_tensors="pt")
embeddings = retriever.get_image_features(**inputs)

# Store `embeddings` in vector DB (e.g., Vespa, FAISS) ...

# Retrieve top-k similar pages → pass to VQA
vqa_model = PaLIFinQA.from_pretrained("google/pali-3b-finqa")
vqa_proc = PaLIProcessor.from_pretrained("google/pali-3b-finqa")

vqa_inputs = vqa_proc(text=query, images=image, return_tensors="pt")
output_ids = vqa_model.generate(**vqa_inputs)
answer = vqa_proc.decode(output_ids[0], skip_special_tokens=True)

print("Answer:", answer)
```

Found this useful?

Save for later

Follow me

Repost Share it

Markus Kuehnle