

Onderzoek: waarom LLM-agents documenten plat slaan (VCA-tool casus)

Datum: 2026-08-05

Aanleiding: bij de VCA-tool legde de agent over 33 verschillende formulieren één en hetzelfde sjabloon (goed/fout/nvt). Elk origineel was anders: reglement, keuzeformulier, registratie.

De agent begreep niet wat elk document **wilde zijn**.

Vraag 1. Waarom platte interpretatie?

Bevindingen.

- **LLM's werken als mode-collapsers.** Onderzoek naar output-diversiteit laat zien dat een model bij herhaalde, vergelijkbare taken terugvalt op één dominant antwoordpatroon ("diversity collapse"). Zodra de eerste paar documenten in een batch één sjabloon krijgen, wordt dat sjabloon het referentiepunt voor de rest. Zie [The Price of Format: Diversity Collapse in LLMs \(arXiv\)](#) en de EMNLP-versie.
- **Context-uitdunning en lost-in-the-middle.** Bij veel documenten in één context verliest het model grip op wat in het midden staat. Prestaties dalen met de hoeveelheid context, zelfs als alles erin past. Document 12 t/m 25 krijgen dan geen verse redenering maar een kopie van het patroon van document 1. Zie [Long Context RAG Performance of LLMs \(arXiv\)](#) en [Lost in the Middle: Why LLMs Struggle With Long Contexts](#).
- **De taak werd nooit expliciet per document gesteld.** De agent kreeg impliciet de taak "maak schermen", niet "bepaal eerst per document wat dit document IS". Zonder die stap vult het model de ontbrekende beslissing in met het bekendste formulier-type uit zijn training (een inspectiechecklist is extreem frequent in trainingsdata rond VCA en veiligheid).
- **Single-pass zonder vergelijking.** Er was geen moment waarop output naast input lag. Een patroonfout over 33 documenten is van binnenuit onzichtbaar: elk scherm ziet er

"netjes" uit. Pas in vergelijking met het origineel valt het op. Dit is het klassieke verschil tussen **plausibel** (klopt de vorm?) en **correct** (klopt de betekenis?).

- **Eén sjabloon is voor het model de goedkoopste oplossing.** Batchverwerking belooft uniformiteit: het is minder "werk" (minder tokens, minder beslissingen) om 33 keer hetzelfde schema te vullen dan 33 keer opnieuw te redeneren. LLM's optimaliseren onbewust op die goedkoopste route als er geen rem op zit.

Vraag 2. Wat zeggen developers op Reddit, HN en fora?

Bevindingen.

- **Classificatie vóór extractie is bij professionele partijen standaard.** Reducto bouwt "classify" expliciet als eerste stap: paspoort krijgt ander schema dan immigratieformulier (Reducto docs). Azure Document Intelligence heeft aparte classificatiemodellen die documenttype bepalen vóór het extractiemodel draait (Microsoft Learn: custom classification models).
- **"Documenttype bepaalt alles" is een bekend Reddit-inzicht.** Een bouwer van een extractieplatform vat het samen: classificatie routeert elk document naar het juiste pad, en je hebt een fallback nodig omdat classificatie ook weleens faalt (Build vs Buy Document Extraction, met Reddit-citaat). Talonic gebruikt zelfs een ontologie van 529 documenttypes om te bepalen welk schema en welke validatieregels gelden (Talonic vs Reducto).
- **OCR/layout-laag en redeneerlaag scheiden.** In productiepijplijnen doet een dedicated OCR-tool (Textract, Azure, Tesseract) het leeswerk; de LLM redeneert over het resultaat. "Never ask Claude to OCR and reason in one shot" (Redwerk: Claude document processing playbook).
- **Golden dataset + eval vóór elke wijziging.** Redwerk noemt als belangrijkste fix: bouw dag 1 een golden dataset van 30 tot 50 gelabelde documenten en draai die vóór elke

prompt-wijziging opnieuw. Zonder eval merk je regressies pas in productie (zelfde bron als hierboven).

- **Human-in-the-loop is geen luxe maar architectuur.** Unstract documenteert HITL als vast onderdeel: mens verifieert extractieresultaten (Unstract HITL).
StackAI beschrijft de volledige keten classificatie, extractie, validatie, menselijke review als de norm voor enterprise (StackAI document processing guide).
- **Academisch bevestigd:** informatie-extractie uit layout-rijke documenten is een eigen ontwerpruimte; één generieke aanpak voor alle layouts werkt niet (arXiv: IE Design Space for Layout-Rich Documents).

Vraag 3. Concrete veranderingen in onze setup

In volgorde van waarde (zie ook het veranderplan onderaan):

1. **Typeclassificatie vóór conversie, per document.** Eerste stap van de pipeline beantwoordt één vraag: "wat wil dit document zijn?" Uitkomst: reglement / keuzeformulier / registratieformulier / checklist / overig. Dit wordt vastgelegd als metadata bij het document.
2. **Eigen schema per type, afgedwongen.** Elk type krijgt een vast scherm-sjabloon. De conversiestap mag ALLEEN het schema van het geclassificeerde type gebruiken. Het goed/fout/nvt-sjabloon is dan één optie onder meerdere, niet de default.
3. **Eén document per run, geen batch van 33.** Elk document krijgt zijn eigen context en eigen redenering. Dit voorkomt patroonbesmetting van document 1 naar 33 en houdt de context klein en scherp.
4. **Visuele zelf-QA per document.** Na conversie: screenshot of render van origineel naast het digitale scherm, en een tweede pass (andere agent of mens) beantwoordt: "doet dit scherm wat het origineel vraagt?" Fail = terug naar stap 1.
5. **Review-checkpoint met mens voor oplevering.** Bij 33 documenten is één review-rondgang met origineel en scherm naast elkaar goedkoop en vangt precies dit soort systematische fouten.

6. **Golden set van 5 tot 10 representatieve formulieren** (één per type) als vaste test.
Elke pipeline-wijziging draait eerst op deze set.
-

Vraag 4. De kernvraag: welke ENKELE stap had DEZE fout gevangen?

De naast-elkaar-vergelijking: origineel document naast het digitale scherm, vóór acceptatie, uitgevoerd door iets of iemand die niet de bouwer was.

Redenering: alle andere oorzaken (mode collapse, context-uitdunning, training-bias) zijn intern voor het model en dus van binnenuit onzichtbaar. Maar de fout was **visueel triviaal** te zien: 33 verschillende origineelen, 33 identieke schermen. Zelfs één review van één willekeurig document uit de batch (bijvoorbeeld het PMO-keuzeformulier naast zijn goed/fout/nvt-scherm) had direct getoond dat het scherm niet doet wat het origineel vraagt. De fout was niet subtiel, hij was alleen nooit bekeken.

Deze stap is ook de goedkoopste: geen nieuwe modellen, geen extra pijn. Alleen de regel "niets is klaar tot origineel en resultaat naast elkaar zijn gelegd en goedgekeurd".

Veranderplan VCA-pipeline (volgorde van waarde)

1. **Review-checkpoint invoeren (de kernstap).** Per document: origineel naast scherm, goedkeuring door mens of tweede agent. Dit vangt deze foutklasse volledig af, ook zonder verdere verbeteringen.
2. **Typeclassificatie als verplichte eerste stap.** Geen conversie zonder vastgelegd documenttype.
3. **Sjabloon keuze koppelen aan type.** Verwijder het default-sjabloon; type bepaalt schema.
4. **Batch uit, per-document runs aan.** Max 1 document per conversierun.
5. **Golden set bouwen** uit de 33 formulieren (de juiste interpretatie per document als referentie) en als regressietest gebruiken.

Bronnen

1. <https://arxiv.org/html/2505.18949v1>
 2. <https://aclanthology.org/2025.findings-emnlp.836.pdf>
 3. <https://arxiv.org/pdf/2411.03538>
 4. <https://pristren.com/blog/lost-in-middle-attention-paper/>
 5. <https://docs.reducto.ai/classify/overview>
 6. <https://learn.microsoft.com/en-us/azure/ai-services/document-intelligence/train/custom-model?view=doc-intel-4.0.0>
 7. <https://imagetotable.ai/blog/build-vs-buy-document-extraction>
 8. <https://talonix.com/vs/reducto>
 9. <https://redwerk.com/blog/claude-document-processing-playbook/>
 0. <https://unstract.com/blog/human-in-the-loop-hitl-for-ai-document-processing/>
 1. <https://www.stackai.com/insights/ai-for-enterprise-document-processing-ocr-end-to-end-workflow-best-practices-and-2026-guide>
 2. <https://arxiv.org/html/2502.18179v1>
-

Bijlage: 3 direct toepasbare fleet-regels (aanvulling 5-8, op verzoek van Conductor)

Deze drie regels passen het onderzoek toe op hoe onze vloot werkt. Ze zijn zo geschreven dat elke bot ze letterlijk kan opvolgen.

Regel 1 — Eerst soort bepalen, dan pas omzetten.

Bij elk document (formulier, PDF, contract, brief): benoem EERST in één zin wat het document WILT zijn (reglement om te ondertekenen / keuzeformulier / registratie / checklist / informatieblad). Die classificatie bepaalt het scherm. Geen classificatie = geen conversie. Voorkomt dat de agent naar zijn bekendste sjabloon grijpt.

Regel 2 — Eén voorbeeld visueel goed laten keuren vóór de batch.

Bij batch-werk (meer dan 1 document/foto/clip tegelijk): lever EERST één compleet voorbeeld

op, leg het naast het origineel, en laat het goedkeuren (door de opdrachtgever of een tweede agent). Pas bij "goed" de rest. Dit is dezelfde regel als onze "1 test → OK → batch" voor foto's en video's — voortaan ook verplicht voor documenten.

Regel 3 — Klaar is pas klaar na de naast-elkaar-check.

Niets heet klaar tot het resultaat naast de bron is gelegd en goedgekeurd. Bij documenten: screenshot of tekst van origineel naast het digitale scherm. Bij uitzondering mag de maker zelf checken, maar bij batch ALTIJD iets/iemand anders (tweede agent of de opdrachtgever). Deze stap had de 33-identieke-schermen fout direct gevangen.