DocAI: Detecting and Localizing AI-Edited Content in Documents

Anonymous authors
Paper under double-blind review

Abstract

Document-level detection of AI-generated content is largely unaddressed, and off-the-shelf tools do not fill the gap: across nine AI-text and AI-image detectors, and both ways of composing them—score the whole page, or segment it and score each region—no configuration exceeds AUROC 0.66. The task demands *dedicated* tools and the data to build them; we provide both. **DocAIgen** edits real documents into large-scale corpora with region-level ground truth, from which we release a $42{,}503$-page benchmark. **DocAIdet** is a document-native Vision-Language Model that reads the page as a whole, jointly detecting and segmenting AI-generated regions. It reaches page-level AUROC 0.944 in-generation and 0.929–0.970 on two unseen generators (0.999 at document level), with Dice 0.895 and 0.699–0.863 respectively; the strongest piecewise baseline, retrained on our own data, reaches 0.874 AUROC and 0.270 Dice on the harder unseen generator, and the margins are significant with non-overlapping 95% intervals. We release the engine, the benchmark and the detector.[1,2]

1 Introduction

AI-generated content (AIGC) now accounts for a large share of what circulates online (Imperva, 2025; Merriam-Webster, 2025), and telling it from human work has become a condition of digital trust (Muzumdar et al., 2025; Landymore, 2025). Forensic research has followed the generators closely—deepfake faces, synthetic images, LLM-written text, lately audio, video, code and tables—yet one ubiquitous data type has been overlooked: the document, the second most common file type on the web after HTML (Johnson, 2021; Common Crawl Foundation, 2025). The threats are concrete: AI-written passages in scientific papers pollute the record. Document AI (Cui et al., 2021) meanwhile supplies models that read a page as a whole, exactly what a forensic method for documents can be built on.

Since a document is a compound of elements, the obvious alternative is to compose the detectors that already exist for each of them; our first finding is that this does not work. A typeset page carries no signal-level trace for an image detector, an in-context edit is too small a share of the page for a text detector, and any such pipeline must first cut the page into fragments, leaving each detector only the evidence that fits inside one (Sec. 3). We build that pipeline with nine off-the-shelf detectors, in both of its readings—score the whole page, or parse it and score every text block and figure crop—and no configuration exceeds AUROC 0.66. Retrained on our own data it becomes a genuine competitor at 0.874, and our single document-native model still beats it on both detection and localization.

We therefore address the two barriers that have kept the problem open. The first is the absence of data: **DocAIgen** parses a real document with Docling (Deep Search Team, 2024), replaces a controlled subset of its components with generated content, re-exports every page through the same renderer, and records each edited region as ground truth. The second is the absence of detectors built for documents: **DocAIdet** is a Vision-Language Model (VLM) that projects the entire page into a joint visual–semantic space and returns a page-level verdict with a pixel-level mask (Fig. 1). Our contributions are the task and the evidence that it needs dedicated tools; DocAIgen and a $42{,}503$-page benchmark with exact region-level ground truth, released *with the engine that generates it* so that

[1] https://anonymous.4open.science/r/AIdocFactory-C301
[2] https://anonymous.4open.science/r/AIdocDetector-A637

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

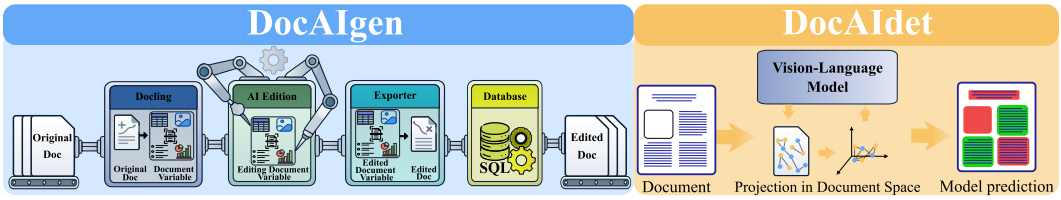


Figure 1: **Overview.** DocAIgen (left) parses a document into components, edits a controlled subset with generative models, re-exports every page—edited or not—through the same pipeline and records each edited region as ground truth. Its corpora train DocAIdet (right), a document-native VLM that reads the rendered page as a whole and returns a page-level verdict and a pixel-level mask of the AI-generated regions.

the corpus can be regrown rather than expire with its generators; DocAIdet, which jointly classifies and segments and beats the strongest baseline by margins whose 95% intervals do not overlap; and an evaluation covering unseen generators on unseen documents, learning-free localization floors, a single-stream ablation, a backbone swap at matched budget, a shortcut audit and, in the appendix, document-level aggregation, laundering robustness, calibration and paired significance tests.

2 Related Work

AI-image and AI-text detection. Image detectors seek generator-specific artifacts in the frequency or spatial domain (Frank et al., 2020; Corvi et al., 2023; Tan et al., 2024), in representational stability under noise (He et al., 2024), or through ViT features (Heo et al., 2023); text detectors are statistical (Solaiman et al., 2019; Vasilatos et al., 2023; Wu et al., 2023) or fine-tuned transformers (Devlin et al., 2019; Liu et al., 2019; Hu et al., 2023), which consistently do better (Wu et al., 2025). Sec. 7.1 evaluates five deployed image detectors—the Swin-based SDXL detector (Organika, 2024) and the Swin-V2 diffusion detector (Haywood, 2024), SigLIP (Ateeqq, 2025) and ViT (Dima806, 2024) classifiers, and the training-free RIGID (He et al., 2024)—and four text detectors: the GPT-2 output detector (Solaiman et al., 2019), the ChatGPT detector, a RoBERTa model trained on ChatGPT responses (Guo et al., 2023), a commercial AI-text detector (Fakespot Future AI Research, 2025) and RADAR (Hu et al., 2023), adversarially trained against paraphrase.

Document AI and manipulation grounding. Our detector is built on the field that models documents in the first place: LayoutLM (Xu et al., 2020), Donut (Kim et al., 2022), DocLLM (Wang et al., 2024), and document-conversion models such as Docling (Deep Search Team, 2024) and SmolDocling (Nassar et al., 2025) consume the page directly and jointly represent its text, layout and visual structure. None is forensic, but they can supply useful representations, and Sec. 8.2 measures how useful they are. Closer to our localization objective are models such as HAMMER (Shao et al., 2024) and TruFor (Guillaro et al., 2023), the latter evaluated in Sec. 7.1, and AI-edited-image datasets such as SID-Set (Huang et al., 2025) and FragFake (Sun et al., 2025). All target *photographic* content, whose signal stems from camera pipelines and low-level generative artifacts; an AI edit in a document replaces a paragraph, a table or a figure on a re-typeset page, which removes that evidence. No prior dataset contains AI-generated content in documents, and without one no model for the task could be trained or even evaluated.

3 Background and Motivation

The task. The input is a document page; the output is a label $\hat{y} \in \{0, 1\}$ and a pixel-level mask $\hat{\mathbf{M}}$ of the AI-generated regions. A page is positive if at least one parsed component—paragraph, table, figure, caption—was replaced by generated content of the same type and role, and the mask's target is the union of the bounding boxes of those components. Two properties matter: the edit is a *semantic substitution* inside an otherwise authentic page, not a manipulation of pixels, and the page is re-typeset afterwards, so edited and untouched components are rendered by the same engine.

Why the signal-level trace is absent. Identity-document forensics and AI-image detection both seek a *signal-level trace*, a footprint left in the pixels by acquisition or synthesis (Ferrara et al., 2012; Guillaro et al., 2023). A born-digital page breaks that assumption: it is typeset, not captured or synthesized from noise, and a replaced paragraph is re-rendered by the same engine. No sensor noise, resampling fingerprint or compression history separates the edited region from the untouched one; the trace does not merely weaken, it does not exist. Every method in Table 1 that seeks such a trace—the AI-image detectors, on the whole page and on figure crops, and the manipulation localizer TruFor—is at chance on our pages (Sec. 7.1).

What the document supplies instead. A document combines textual, layout and visual elements to convey an intent (Briet, 1951; Buckland, 1997), and an edit leaves evidence of two kinds, both at page scale: the statistical signature of the regenerated content—the cue a text detector seeks, but on a fragment too short to settle the question—and how that content sits among everything else: not its subject, since edits stay on topic (Appendix E.1), but its register, density, phrasing, and the way it discharges the role its position assigns it among the text, headings, captions, tables and figures around it.

Why the page cannot be taken apart. A pipeline of existing detectors must cut the page into pieces and hand each to a model that sees its fragment and nothing else. What is intrinsic to a fragment survives the cut and is worth something: retrained on our data, such a system is the best method after ours. Everything else is lost: a paragraph that is inconclusive alone is often conclusive beside its neighbours, and no fusion rule over fragment scores restores a comparison the fragmentation has destroyed; nor has such a pipeline anywhere to put layout, reading order and component roles. Localization, moreover, falls to the extraction stage such a pipeline needs, since a score attaches to a string and a string has no coordinates. A piecewise system only ever sees a bag of parts, and the results show the deficit that predicts (Sec. 7.1).

4 DocAIgen

The primary obstacle is the absence of data: to the best of our knowledge, no dataset of AI-generated content in documents exists, not even a small one. The first pillar of our work is therefore DocAIgen, a pipeline of four modules (Fig. 1). **Docling parsing** (Deep Search Team, 2024) extracts every component of the document and its attributes—titles, paragraphs, images, tables, footnotes, references—into a hierarchical *Document Variable* that retains everything needed to reconstruct it (Appendix G.5). **AI edition** replaces selected components of the Document Variable, routing each to a submodule by component type; the generator for each type is user-specified, by Hugging Face identifier (Hugging Face Transformers Team, 2026) or as a local model, under configurable selection parameters (Appendix G.6). Crucially, only the semantic content is modified (Appendix G.5); all structural and layout metadata is left untouched. Finally the **exporter** renders the edited Document Variable into a new document. Original and generated components pass through the identical pipeline (original components simply pass through the AI-edition module unchanged), so the sole difference between an authentic and a modified component is its intrinsic content—verified qualitatively and with four low-level forensic tests in Appendices G.3 and G.4. In parallel, the **database** module records which components were altered and their precise bounding boxes—the ground truth for DocAIdet—together with the generated content itself (Appendices G.1, G.2).

This is what distinguishes our release from a dataset. A benchmark built from a fixed set of generators expires with them; because the engine ships with the corpus, ours can instead be regrown, ground truth included, with no annotation effort.

5 DocAIdet

DocAIdet analyzes a page as a whole and produces a binary label and a segmentation mask (Fig. 2). A page is tiled into N local patches plus one global view, giving $N{+}1$ inputs to the vision encoder; C is the number of spatial tokens per input, $D_v{=}768$ the width of vision-encoder features, $D_t{=}576$ that of the language-decoder hidden states, and D_{proj} that of the shared space. Five components map a page to a multi-scale visual pyramid and a semantic stream, align them, and decode them into a label and a mask: (i) a *vision stream* tapped at multiple encoder depths; (ii) a *semantic stream*

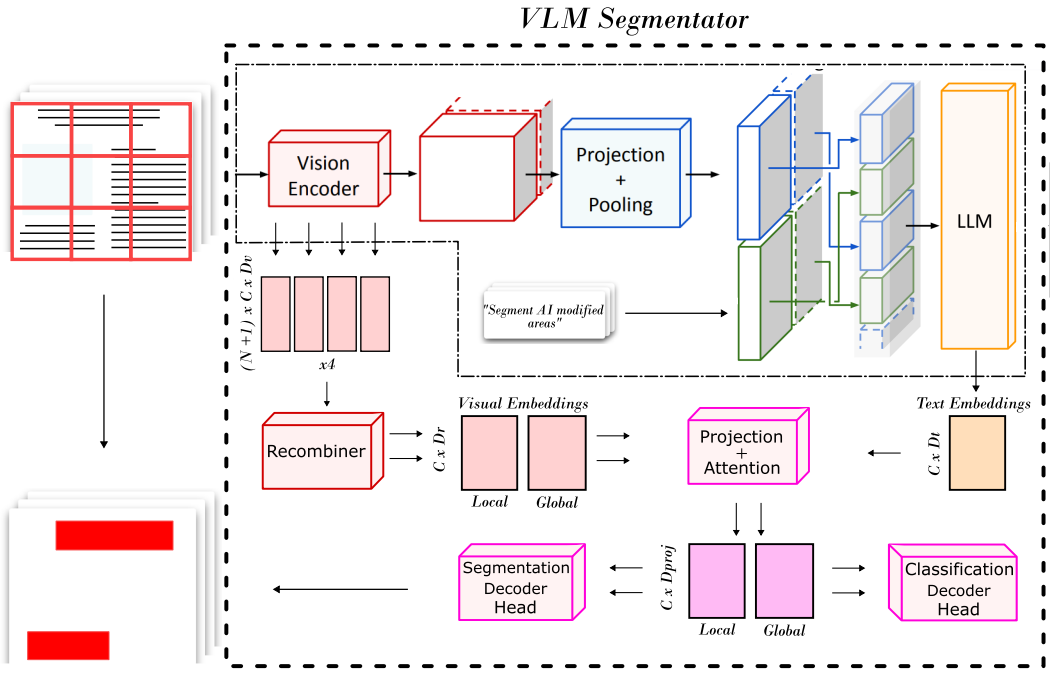


Figure 2: DocAIdet. The page is processed as a whole and decoded into a page-level label and a pixel-level mask of the synthetic regions. Tensor shapes are annotated on the arrows.

from the language decoder's hidden states; (iii) a *recombiner* that restores the spatial geometry the tiled encoder destroys; (iv) a *gated cross-modal fusion* block that lets each spatial position decide how much semantic evidence to admit; and (v) two *decoder heads*. The vision and semantic streams are initialized from SmolDocling-256M (Nassar et al., 2025), a document-conversion model whose encoder and decoder already represent page structure, layout and reading order. The full model is trained end to end (Sec. 6.2): the streams are reshaped by Low-Rank Adaptation (Hu et al., 2022) and the other three components are trained from scratch. Sec. 8.1 shows that neither stream learns the task alone, and Sec. 8.2 that the initialization is worth 0.18 Dice.

Vision and semantic streams. The vision stream is read from the encoder at layers 3, 7 and 11: shallower layers preserve the high-resolution detail that supports precise boundaries while deeper layers encode global context, and fusing them lets the segmentation head place an edited region accurately at page resolution. The semantic stream is the language decoder's hidden states for the same page, which carry what the page says and—from a backbone pretrained for document conversion—the structural role in which it says it.

Recombiner. The vision encoder returns $N{+}1$ independent feature maps, in which the spatial relationship between tiles has been destroyed. Writing $\mathbf{V}^{(\ell)} \in \mathbb{R}^{(N+1)\times D_v\times h\times w}$ for the features tapped at layer ℓ, with the tiling arranged as a $G_h \times G_w$ grid so that $N = G_h G_w$, the recombiner discards the global view and re-tiles the local ones into a single spatially coherent map,

$$\big[\mathcal{R}(\mathbf{V}^{(\ell)})\big]_{:,\,ih+p,\,jw+q} \;=\; \mathbf{V}^{(\ell)}_{j+iG_w,\,:,\,p,\,q}, \tag{1}$$

for $0 \le i < G_h$, $0 \le j < G_w$, $0 \le p < h$, $0 \le q < w$, giving $\mathcal{R}(\mathbf{V}^{(\ell)}) \in \mathbb{R}^{D_v\times G_h h\times G_w w}$; the global view is kept separately as a page-level context map. Applied at $\ell \in \{3, 7, 11\}$, Eq. 1 yields the multi-scale pyramid the heads consume. The operation is parameter-free.

Cross-modal fusion. Let $\mathbf{v}^{(\ell)} \in \mathbb{R}^{HW\times D_v}$ be the recombined map at level ℓ flattened over space, and $\mathbf{s} \in \mathbb{R}^{L\times D_t}$ the language-decoder hidden states for the same page. The semantic stream is projected into the vision width, $\tilde{\mathbf{s}} = \mathbf{s}\mathbf{W}_s$ with $\mathbf{W}_s \in \mathbb{R}^{D_t\times D_v}$, and each spatial position attends

over it with 8 heads:

$$\mathbf{a}^{(\ell)} \;=\; \mathrm{MHA}\big(\underbrace{\mathbf{v}^{(\ell)}}_{\text{query}}, \underbrace{\tilde{\mathbf{s}}}_{\text{key}}, \underbrace{\tilde{\mathbf{s}}}_{\text{value}}\big). \tag{2}$$

Attention alone would force every position to absorb semantic content whether or not any text lies under it, so the residual is gated per position by a bottleneck MLP,

$$\mathbf{g}^{(\ell)} \;=\; \sigma\big(\,\mathrm{ReLU}(\mathbf{v}^{(\ell)}\mathbf{W}_1)\,\mathbf{W}_2\big) \;\in\; (0,1)^{HW\times 1}, \qquad \mathbf{z}^{(\ell)} \;=\; \mathrm{LayerNorm}\,\big(\mathbf{v}^{(\ell)} + \mathbf{g}^{(\ell)} \odot \mathbf{a}^{(\ell)}\big), \tag{3}$$

where $\mathbf{W}_1 \in \mathbb{R}^{D_v \times D_v/4}$ and $\mathbf{W}_2 \in \mathbb{R}^{D_v/4\times 1}$. Because $\mathbf{g}^{(\ell)}$ is computed from $\mathbf{v}^{(\ell)}$ alone, the model decides *where* semantics are worth consulting before it consults them; three such blocks are applied to the local pyramid and three to the global context map.

Heads. Both heads read $\{\mathbf{z}^{(\ell)}\}$ restored to $\mathbb{R}^{D_v\times H\times W}$ and stay deliberately light. The segmentation head projects each level to 256 channels with 1×1 convolutions, refines with depthwise-separable 3×3 convolutions under skip connections, applies an FFN, and predicts logits $\hat{\mathbf{M}} \in \mathbb{R}^{H_0\times W_0}$ at page resolution with a final 1×1 convolution. The classification head pools each stream by learned spatial attention,

$$\mathbf{u} \;=\; \sum_{k=1}^{HW} \alpha_k \mathbf{z}_k, \qquad \boldsymbol{\alpha} \;=\; \mathrm{softmax}_k\big(\mathbf{z}\mathbf{w}_a\big), \tag{4}$$

so that the decision is dominated by informative positions rather than by an average over mostly-pristine area, then fuses the local and global pooled vectors by cross-attention before a linear classifier gives $\hat{y}$. Appendix F.5 gives the layer-by-layer specification and parameter counts.

6 Experimental Setup

6.1 Benchmark

The benchmark is built from PDFs, the vast majority of online documents (Johnson, 2021; Common Crawl Foundation, 2025), sourced from arXiv, whose categorization by field ensures variability and whose peer-reviewed subset serves as quality control. By arXiv's temporal metadata, 80% of the articles predate 2021, so the pristine class largely predates widespread LLM use; the remaining 20% are post-2021 but pre-2025, to capture evolving layout and writing conventions (Bizzoni et al., 2020) at a marginal mislabeling risk that the peer-review filter further limits. We collect 901 real documents and generate 1,299 AI-edited counterparts, yielding 42,503 rendered pages. A document is edited by replacing a subset of its parsed components—paragraphs, tables, figures—with generated content of the same type and role, so an edited page mixes generated and original components; the recorded bounding box of every replaced component is the ground truth.

No rendering shortcut. Every page, edited or pristine, is produced by the *same* operation: parse the source PDF into the Document Variable, then re-export it through the same renderer. A pristine document is parsed and re-rendered exactly like an edited one, with no component replaced, so the export is a constant function of the class and carries no discriminative signal. What separates the classes is *who created* a component, with everything else held fixed, including the subject, since the generator is conditioned on the component it replaces (Appendix E.1); Sec. 8 tests both claims.

In-Generation and Cross-Generation. A detector that has learned one generator's idiom rather than the general signature of machine-written content fails on a new generator, the standard failure mode in AIGC detection (Ojha et al., 2023; Wu et al., 2025). We therefore split by generator: the training and *In-Generation* corpora are edited with Llama-3.2-3B for text components and SD-Turbo for image ones, and the *Cross-Generation* corpora with Qwen3-4B and Granite-4.0-h for text components and Stable-Diffusion-v1.5 for image ones, which appear nowhere in training (Appendix G.6).

6.2 Training

The training set comprises 16,027 pages drawn exclusively from the In-Generation corpus (67% AI-edited, 33% pristine), plus 3,434 pages for validation and 3,435 for testing. Training took about

ten days on one NVIDIA H100 (94 GB) in `bfloat16` with batch size 12, in two phases: Phase 1 optimizes the entire architecture except the classification head with a combined focal and Dice loss ($2\,\mathcal{L}_{\text{focal}} + \mathcal{L}_{\text{Dice}}$, following SAM); Phase 2 freezes those weights and trains only the classification head with weighted cross-entropy for 40 epochs, although it converges within a few: Phase 2 is a rapid alignment step, since the Phase-1 embeddings already encode the presence of AIGC. Losses, hyperparameters and learning curves are in Appendices F.1 and F.2.

6.3 Evaluation Sets and Protocol

We evaluate on class-balanced corpora. The *In-Generation* set is a balanced subsample (2,190 pages) of the test split above, edited by the training generators. The *Cross-Generation* corpora are edited by generators never seen in training (Appendix G.6.1); the larger and less favourable of the two, edited by Qwen3-4B, gives the headline Cross-Generation number. One protocol governs everything: the same code scores every method on the same pages; classification uses a fixed threshold of 0.5, tuned on no test set; and confidence intervals are 95% percentile bootstrap intervals over pages (10,000 resamples), paired across models (Appendix E.3). Localization is reported as Dice macro-averaged per page over the *AI-edited* pages of each corpus, since an empty prediction scores 1.0 against a pristine page's empty target and averaging such pages in would mix delineating a region with correctly producing none; pristine pages carry every page-level number, and their masks are reported separately (Sec. 7.2).

7 Results

7.1 Can Existing Methods Solve This Task?

Table 1 collects every alternative we could run under one protocol (classification detail in Appendix E.5). "Piecewise baseline" hides two systems. *Whole-page*: the page text goes to a text detector and the rendered page to an image detector, and the verdicts are fused; the text is taken from the very PDFs the images were rendered from, the most favourable extraction possible. *Region-level*: the page is parsed first (Docling), each layout text block going to the text detector and each figure crop to the image detector—stronger, with more opportunities to fire, but downstream of a parser as well as two detectors. Both use the same components and fusion rule (a page scores as the maximum of its parts).

Image-level detection has nothing to detect, at either granularity. We run the five image detectors of Sec. 2. On the *whole page*, all five sit inside $[0.485, 0.524]$ AUROC on both corpora—chance, as Sec. 3 predicts. The *figure crops* are decisive, since the typesetting argument does not cover a figure: a generation trace, if it survived anywhere, would survive in a regenerated raster. It does not—the four detectors admitting a crop-level score reach only $[0.457, 0.562]$, since the figures are plots and diagrams, off-distribution for natural-image detectors, and the crop comes from the rasterized page rather than the generator's output file. TruFor (Guillaro et al., 2023), a state-of-the-art manipulation localizer, is likewise at chance (0.520 AUROC, Dice 0.042), and RIGID, which cannot be dismissed as distribution shift since it knows only representational stability, carries no label information either.

Text-level detection is diluted, and segmenting the page trades dilution for over-firing. Four off-the-shelf AI-text detectors read the concatenated page text and reach 0.512–0.659 AUROC In-Generation and 0.478–0.620 Cross-Generation, the best (RADAR) 0.29–0.31 behind ours, for a structural reason: only a fraction of a page is edited, and the surrounding original text dominates the score. Scoring each layout block independently removes that dilution, lifting the detector of Fakespot Future AI Research (2025) from 0.566 to 0.615 In-Generation, but introduces the opposite failure: the maximum over a dozen blocks of academic prose flags most genuine pages, specificity falling to 0.359 and 0.240. No off-the-shelf configuration, at either granularity and with any of the nine detectors, exceeds AUROC 0.66. Combining the two modalities changes nothing: the page-level combination reaches 0.566 and 0.505, the region-level one 0.614 and 0.562, each within 0.001 of its own *text* arm, since the image detector scores lower on essentially every page—the piecewise solution is a text detector with an inert second channel.

Table 1: Every method we could run, on the two benchmark test sets, under one protocol. "Reads" is the modality consumed and "Applied to" the granularity; "Loc." marks methods that emit a spatial mask. Dice is macro-averaged over edited pages (Sec. 6.3); for the trained piecewise rows the mask comes from their SegFormer-B0 segmentation arm (Sec. 7.1). †: scored on the pages containing a figure (787 In-Generation, 721 Cross-Generation), or, for TruFor, on a balanced 1,000-page subset. —: not applicable or not run. Baselines are scored on a balanced 3,000-page Cross-Generation subsample on which our AUROC differs by less than 0.001 (Appendix E.5); detectors are cited in Sec. 2.

				In-Generation		Cross-Generation	
Method	Reads	Applied to	Loc.	AUROC	Dice	AUROC	Dice
Off-the-shelf, not trained on our data							
SDXL detector	image	whole page	✗	0.497	—	0.499	—
Swin-V2 AI-image detector	image	whole page	✗	0.518	—	0.502	—
SigLIP AI-image detector	image	whole page	✗	0.512	—	0.524	—
ViT AI-image detector	image	whole page	✗	0.497	—	0.492	—
RIGID	image	whole page	✗	0.513	—	0.485	—
SDXL detector	image	figure crops	✗	0.509†	—	0.472†	—
Swin-V2 AI-image detector	image	figure crops	✗	0.562†	—	0.524†	—
SigLIP AI-image detector	image	figure crops	✗	0.513†	—	0.457†	—
ViT AI-image detector	image	figure crops	✗	0.476†	—	0.488†	—
TruFor	image	whole page	✓	—	—	0.520†	0.042†
AI-text detector	text	whole page	✗	0.566	—	0.504	—
AI-text detector	text	text blocks	✗	0.615	—	0.563	—
RADAR	text	whole page	✗	0.659	—	0.620	—
RADAR	text	text blocks	✗	0.567	—	0.565	—
ChatGPT detector	text	whole page	✗	0.563	—	0.594	—
ChatGPT detector	text	text blocks	✗	0.591	—	0.579	—
GPT-2 output detector	text	whole page	✗	0.512	—	0.478	—
GPT-2 output detector	text	text blocks	✗	0.503	—	0.485	—
Piecewise, page-level fusion	text+image	whole page	✗	0.566	—	0.505	—
Piecewise, region-level fusion	text+image	blocks + figures	✗	0.614	—	0.562	—
Frontier MLLMs, prompted zero-shot							
GPT-5.2	text+image	whole page	✗	0.621	—	0.612	—
Gemini-3-flash	text+image	whole page	✗	0.821	—	0.823	—
Claude-opus-4.5	text+image	whole page	✗	0.676	—	0.664	—
Trained on our training split							
SegFormer-B0	image	whole page	✓	0.750	0.483	0.643	0.270
AI-text detector, fine-tuned	text	whole page	✗	0.809	—	0.852	—
AI-text detector, fine-tuned on blocks	text	text blocks	✗	0.726	—	0.820	—
Piecewise, page-level fusion	text+image	whole page	✓	0.835	0.483	0.852	0.270
Piecewise, region-level fusion	text+image	blocks + page	✓	0.863	0.483	0.874	0.270
Ours (DocAIdet)	text+image	whole page	✓	**0.944**	**0.895**	**0.929**	**0.699**

The failure is not merely domain mismatch. We rebuild the whole pipeline from components trained on our own split, at both granularities; each pair of numbers below is In-Generation and Cross-Generation. Fine-tuning the text detector on whole pages lifts its AUROC to 0.809 and 0.852, but only by flagging most genuine pages (specificity 0.401 and 0.443). Training it instead on individual layout blocks, each labelled by whether it was regenerated, ranks lower (0.726 and 0.820 AUROC) at a far more usable operating point (specificity 0.961 and 0.923). Fusing each with SegFormer-B0 reaches 0.835 and 0.852 AUROC at page level and 0.863 and 0.874 at region level, the best baseline in the table (a learned fusion fitted on one corpus and evaluated on the other lands at 0.840 and 0.817). A fully trained piecewise system is thus a respectable detector, but not ours: at region level it trails by 0.081 and 0.055 AUROC while running two models and a parser, operates at specificity 0.439 and 0.357 against our 0.882 and 0.884, and localizes only as well as its SegFormer arm (0.483 and 0.270 Dice against our 0.895 and 0.699). Since every component saw our data, what it lacks is not adaptation but access to the page as a whole—and the deficit falls where Sec. 3 predicts: under the generator shift, 0.055 AUROC on *whether* and 0.430 Dice on *where*.

Generic architectures and frontier MLLMs. SegFormer-B0 trained directly on our annotations reaches 0.643 AUROC and 0.270 Dice under the generator shift, losing 0.107 AUROC and 44% of its Dice where we lose 0.015 and 22%. Prompted multimodal models are the strongest alternative (Gemini Team et al., 2023; Zhang et al., 2025; Li et al., 2024): Gemini-3-flash reaches 0.823 Cross-Generation, 0.106 behind ours, while GPT-5.2 and Claude-opus-4.5 are so biased toward "real" that

precision 0.917 comes at recall 0.148. None produces a mask, and all cost orders of magnitude more compute per page than our 265M-parameter model.

Every baseline ran on exactly our pages, so the margins are tested pairwise: every McNemar test falls below any conventional threshold and no bootstrap interval on the AUROC gap crosses zero (Appendix E.3). At 265M parameters and 26 pages per second on one GPU, ours is also the cheapest to run (Appendix E.4).

7.2 Localization, and Generalization to Unseen Corpora

Our detector reaches AUROC 0.944 In-Generation and 0.929 Cross-Generation, with Dice 0.895 and 0.699 (Table 1). Two learning-free predictors give the floor: the whole page, and every layout component Docling finds (Appendix A). Over all pages our Dice is 0.941, against 0.243 for the layout mask and 0.129 for the whole-page mask. What separates us from them is the ability to mark nothing: on a pristine page we return an empty mask 98.7% of the time, while the layout mask, which covers 43.5% of an average page, scores 0.009 and the whole-page mask 0.000; both fire on every page, so both sit at AUROC 0.5. On edited pages the floors look much stronger (0.478 and 0.258 against our 0.895), because edited regions are layout components and a mask over all of them covers the target; our headline Dice is that stricter number. Detecting and localizing are different problems: the fully trained piecewise pipeline reaches AUROC 0.874 but Dice 0.270, SegFormer-B0 the same 0.270, and TruFor 0.042.

Where it fails. The detector recovers 80.8% of edited regions at 50% coverage, and component area explains the spread: recall falls almost monotonically with size, from 0.91–0.93 on tables and figures to 0.86 on text blocks and 0.03 on page headers and footers, which carry too few words for any statistical judgement (Appendix A). Below roughly one percent of page area an edit becomes hard to detect (Fig. 3), a limit set by the evidence rather than by the architecture.

Unseen generators on unseen documents. Deciding *whether* costs nothing: page-level AUROC is 0.929 on Qwen3-4B and 0.970 on Granite-4.0-h against 0.944 in-generation, and the best competing method reaches 0.874 on the same pages. Deciding *where* drops, but on one generator only: Dice falls to 0.699 on Qwen and stays at 0.863 on Granite. Granite is exactly as unfamiliar as Qwen, so the cause is how one particular generator writes, and the probability map before thresholding says what kind of failure it is (Appendix A.3). The model still knows where the edit is: on 89.8% of edited Qwen pages the single most confident pixel falls inside the edited region, and the probability inside remains about twenty times that outside, as on the other corpora. What changes is confidence: the mean probability inside the target drops by about a quarter, so on one edited Qwen page in eight the mask is empty, and where a mask appears it is correct but covers about four-fifths of the region. Outside the training generator, then, the page verdict remains reliable and the mask is a floor on the edit's extent, consistent with generators differing in how far their signatures sit from human text (Fraser et al., 2025).

8 Analysis and Diagnostics

Shortcut audit. We check that the benchmark cannot be solved without doing the task, from both directions in which a shortcut could exist. Appearance offers none: every method that seeks a pixel-level trace is at chance (Sec. 7.1), and the visual stream alone cannot even fit its training set (Sec. 8.1). Nor does topic: edited pages retain 92.8% of the lexical similarity an untouched page has to its own document, and one edited page in a thousand is closer to a random document than to its own (Appendix E.1). Appendices B–E add what the detector delivers in use: document-level AUROC 0.980 In-Generation and 0.999 Cross-Generation by averaging page probabilities, robustness to JPEG re-compression down to $q{=}50$ (halving the resolution is what hurts), calibration out of the box (ECE 0.073) with a flat false-positive rate across disciplines, and qualitative examples.

8.1 Which Stream Carries the Evidence?

The architecture rests on the claim that an edit is found by reading the page's appearance and its content *together*; the two suspicions are that the model is really a vision model reacting to rendering

artifacts, or really a text detector with a page attached. We train one decoder on the visual pyramid alone and another on the language decoder's hidden states alone, with no cross-attention or fusion (curves in Appendix F.3). The vision-only model never converges—training Dice near 0.3, validation Dice around 0.15 after 50 epochs, failing to fit even the data it is shown—which is not a generalization failure but the absence of a signal to fit, and refutes the rendering-artifact hypothesis directly. The semantic-only model learns quickly and then saturates: validation Dice flattens near 0.5 by the seventh epoch and never moves again. The capability therefore sits in neither stream but in the gated cross-attention binding them (Eqs. 2–3).

A further intervention at inference rules out the remaining explanation. Feeding each page another page's semantics—both streams intact, their correspondence destroyed—collapses AUROC from 0.933 to 0.519 (on the balanced $1{,}000$-page subsample of Appendix C). A model applying a generic stylistic prior over machine-written text would not care whose text it was given; this one compares what the page *says* against what it *looks like*.

8.2 How Much Does the Document-Native Backbone Contribute?

If, as Sec. 3 argued, the task is solved at the semantic and structural level of the page, document pretraining should matter, and most on localization; if not, any VLM of the same size would do. The deciding experiment is a backbone swap with everything else fixed: we retrain the complete pipeline from SmolVLM-256M-Instruct, a general-purpose model of the same architecture family and parameter count with no document-conversion objective in its pretraining, with the same corpus, $40{+}40$ epoch schedule, batch size, learning rates and seed, and no warm start; both arms are scored on the In-Generation test set (Appendix F.4). Pretraining matters: paired over the same pages, the general-purpose backbone loses 0.040 AUROC (paired bootstrap $[0.030, 0.049]$) and 0.182 Dice ($[0.168, 0.195]$; McNemar $p = 1.1{\times}10^{-13}$ on accuracy). And it matters asymmetrically, in the direction predicted: the Dice gap is nearly five times larger in relative terms than the classification gap. Detecting that *something* was regenerated is comparatively portable—the generic VLM reaches AUROC 0.905, above every baseline in Table 1 except ours—but saying *where* needs a representation in which the page is already parsed into components, roles and reading order, built by a document-conversion objective rather than a general image–text one: Dice 0.714 against 0.895. The initialization is not the method, though: SmolDocling transcribes a page and emits no forensic judgement until the architecture around it is trained, whose worth Sec. 8.1 measures.

9 Conclusions and Limitations

We presented DocAIgen, which produces document corpora with exact region-level ground truth, and DocAIdet, a document-native VLM that jointly detects and localizes AI-edited content. Our central finding is that the problem requires dedicated tools: composing off-the-shelf detectors fails at both granularities, and rebuilding every component on our own data yields a respectable detector (0.874 Cross-Generation) that still trails a single document-native model by 0.055 AUROC at less than half its specificity and localizes at 0.270 Dice against our 0.699. What it lacks is access to the page as a whole, and the deficit falls where that predicts: small on *whether*, large on *where*. Neither stream learns the task alone, and a general-purpose VLM of identical size, data and schedule loses 0.182 Dice: document forensics should be built on document models.

The limits are concrete. Small edits are the hard case: page-level recall falls from 0.996 above 15% of page area to 0.428 below 1% (0.315 when the generator is also unseen), the direction in which the model was deliberately built to fail, since a false accusation is the costlier error (Appendix E.6). Localization degrades on one of two unseen generators ($0.895 \rightarrow 0.699$) through a loss of confidence rather than of placement, so outside the training generator the mask is a floor on where the edit is. Resolution, not compression, is the operational constraint (Appendix C). The domain is scientific preprints in English; extending it is a configuration change for the released engine, not a new annotation effort.

Ethics statement

Our work aims to strengthen information integrity by detecting AI-edited documents. Two considerations warrant care. *False accusations:* because incorrectly flagging a genuine document is harm-

ful, we explicitly optimize for high specificity and recommend deploying the detector as a triage aid for human review rather than as an automated verdict; we report calibration and per-domain false-positive rates (Appendix E.2) to support this use. *Dual use of the generator:* DocAIgen can, in principle, be used to produce misleading documents at scale. We judge that the benefit of enabling detection research outweighs this risk, but we release it under a license restricting malicious use and accompanied by the detector and the benchmark, so that generation and detection advance together. The benchmark contains no personal data beyond what is public in arXiv preprints, and its edits are synthetic: it must not be read as evidence about any arXiv author.

Reproducibility statement

The generation engine, the benchmark and the trained detector are released at the anonymous links given in the abstract. Sec. 6 describes corpus construction, splits and the evaluation protocol; Appendix G.6 gives the generation hyperparameters and the identity of every generator; Appendix F.1 the loss functions and Appendix F.2 the LoRA configuration, optimizer and schedule; and Appendix F.5 the stage-by-stage architecture specification with parameter counts.

AI use statement

In this work, we used generative AI tools (large language models) for translating and condensing some parts of the text of the manuscript from a longer internal draft written by the authors. We have not used generative AI tools for generating research ideas, for designing the experiments, or for producing or analyzing any of the reported results. The AI-edited documents that constitute the benchmark are generated by the open-weight models named in Appendix G.6.1: they are the object of study, not authoring tools. All AI-assisted text was reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Ateeqq. AI vs Human Image Detector. `https://huggingface.co/Ateeqq/ai-vs-human-image-detector`, 2025. SigLIP synthetic image detector. Accessed 2026.

Yuri Bizzoni, Stefania Degaetano-Ortlieb, Peter Fankhauser, and Elke Teich. Linguistic variation and change in 250 years of English scientific writing: A data-driven approach. *Frontiers in Artificial Intelligence*, 3:73, 2020. doi: 10.3389/frai.2020.00073.

Suzanne Briet. *Qu'est-ce que la documentation?* EDIT, Paris, 1951.

Michael K Buckland. What is a document? *Journal of the American Society for Information Science*, 48(9):804–809, 1997.

Common Crawl Foundation. Common Crawl MIME type statistics. Common Crawl Repository, 2025. Available online at `https://commoncrawl.github.io/cc-crawl-statistics/plots/mimetypes`.

Riccardo Corvi, Davide Cozzolino, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. Intriguing properties of synthetic images: from generative adversarial networks to diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 973–982, 2023.

Lei Cui, Yiheng Xu, Tengchao Lv, and Furu Wei. Document AI: Benchmarks, models and applications. *arXiv preprint arXiv:2111.08609*, 2021.

Deep Search Team. Docling technical report. Technical report, IBM, 8 2024. URL `https://arxiv.org/abs/2408.09869`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Dima806. AI vs Real Image Detection. https://huggingface.co/dima806/ai_vs_real_image_detection, 2024. ViT synthetic image detector. Accessed 2026.

Fakespot Future AI Research. RoBERTa-base AI Text Detector. https://huggingface.co/fakespot-ai/roberta-base-ai-text-detection-v1, 2025. Accessed 2026.

Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. Image forgery localization via fine-grained analysis of CFA artifacts. *IEEE Transactions on Information Forensics and Security*, 7(5):1566–1577, 2012.

Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. Leveraging frequency analysis for deep fake image recognition. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3247–3258. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/frank20a.html.

Kathleen C Fraser, Hillary Dawkins, and Svetlana Kiritchenko. Detecting AI-generated text: Factors influencing detectability with current methods. *Journal of Artificial Intelligence Research*, 82: 2233–2278, 2025.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

IBM Granite Team. Granite 3.0 language models, October 2024. URL https://github.com/ibm-granite/granite-3.0-language-models/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Fabrizio Guillaro, Davide Cozzolino, Avneesh Sud, Nicholas Dufour, and Luisa Verdoliva. TruFor: Leveraging all-round clues for trustworthy image forgery detection and localization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20606–20615, 2023.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. How close is ChatGPT to human experts? Comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*, 2023.

Sloan Haywood. AI Image Detector (deploy). https://huggingface.co/haywoodsloan/ai-image-detector-deploy, 2024. Swin-V2 synthetic image detector. Accessed 2026.

Zhiyuan He, Pin-Yu Chen, and Tsung-Yi Ho. RIGID: A training-free and model-agnostic framework for robust AI-generated image detection. *arXiv preprint arXiv:2405.20112*, 2024.

Young-Jin Heo, Woon-Ha Yeo, and Byung-Gyu Kim. Deepfake detection algorithm based on improved vision transformer. *Applied Intelligence*, 53(7):7512–7527, 2023.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. RADAR: Robust AI-text detection via adversarial learning. *Advances in neural information processing systems*, 36:15077–15095, 2023.

Zhenglin Huang et al. SIDA: Social media image deepfake detection, localization and explanation with large multimodal model. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, 2025.

Hugging Face Transformers Team. Configuration. https://huggingface.co/docs/transformers/main_classes/configuration, 2026. Accessed: January 19, 2026. Documentation for loading model configurations from Hugging Face identifiers via config files.

Imperva. 2025 Bad Bot Report. Technical report, Thales Group, April 2025. URL https://cpl.thalesgroup.com/sites/default/files/content/campaigns/badbot/2025-Bad-Bot-Report.pdf. Accessed: January 13, 2026.

Duff Johnson. PDF's popularity online. PDF Association, September 2021. URL https://pdfa.org/pdfs-popularity-online/. Based on the July 2021 Common Crawl data.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. OCR-free document understanding transformer. In *European Conference on Computer Vision*, pp. 498–517. Springer, 2022. doi: 10.1007/978-3-031-19815-1_29.

Frank Landymore. Sam Altman says He's suddenly worried Dead Internet Theory is coming true. Futurism, September 2025. URL https://futurism.com/sam-altman-dead-internet-theory. Citing Altman's tweet on LLM-run accounts.

Jian Li, Weiheng Lu, Hao Fei, Meng Luo, Ming Dai, Min Xia, Yizhang Jin, Zhenye Gan, Ding Qi, Chaoyou Fu, et al. A survey on benchmarks of multimodal large language models. *arXiv preprint arXiv:2408.08632*, 2024.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Merriam-Webster. 2025 Word of the Year: Slop. https://www.merriam-webster.com/wordplay/word-of-the-year, December 2025. Accessed: January 13, 2026. Cultural context for AI-generated content saturation.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *International conference on machine learning*, pp. 24950–24962. PMLR, 2023.

Prathamesh Muzumdar, Sumanth Cheemalapati, Srikanth Reddy RamiReddy, Kuldeep Singh, George Kurian, and Apoorva Muley. The Dead Internet Theory: A survey on artificial interactions and the future of social media. *arXiv preprint arXiv:2502.00007*, 2025. URL https://arxiv.org/abs/2502.00007.

Ahmed Nassar, Matteo Omenetti, Maksym Lysak, Nikolaos Livathinos, Christoph Auer, Lucas Morin, Rafael Teixeira de Lima, Yusik Kim, A Said Gurbuz, Michele Dolfi, and Peter W J Staar. SmolDocling: An ultra-compact vision-language model for end-to-end multi-modal document conversion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21972–21983, 2025.

Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24480–24489, 2023.

Organika. SDXL Detector. https://huggingface.co/Organika/sdxl-detector, 2024. Swin-based synthetic image detector. Accessed 2026.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024.

Rui Shao, Tianxing Wu, Jianlong Wu, Liqiang Nie, and Ziwei Liu. Detecting and grounding multi-modal media manipulation and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5556–5574, 2024.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*, 2019.

Zhen Sun et al. FragFake: A dataset for fine-grained detection of edited images with vision language models. *arXiv preprint arXiv:2505.15644v1*, 2025.

Chuangchuang Tan, Huan Liu, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Rethinking the up-sampling operations in CNN-based generative network for generalizable deepfake detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28130–28139, 2024.

Yuchuan Tian, Hanting Chen, Xutao Wang, Zheyuan Bai, Qinghua Zhang, Ruifeng Li, Chao Xu, and Yunhe Wang. Multiscale positive-unlabeled detection of AI-generated texts. In *International Conference on Learning Representations*, 2024.

Christoforos Vasilatos, Manaar Alam, Talal Rahwan, Yasir Zaki, and Michail Maniatakos. HowkGPT: Investigating the detection of ChatGPT-generated university student homework through context-aware perplexity analysis. *arXiv preprint arXiv:2305.18226*, 2023.

Dongsheng Wang, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu. DocLLM: A layout-aware generative language model for multimodal document understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8529–8548, 2024.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. A survey on LLM-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, 51(1):275–338, 2025.

Kangxi Wu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. LLMDet: A third party large language models generated text detection tool. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2113–2133, 2023. doi: 10.18653/v1/2023.findings-emnlp.139.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. LayoutLM: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1192–1200, 2020.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. DialoGPT: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th annual meeting of the association for computational linguistics: system demonstrations*, pp. 270–278, 2020.

Zicheng Zhang, Junying Wang, Farong Wen, Yijin Guo, Xiangyu Zhao, Xinyu Fang, Shengyuan Ding, Ziheng Jia, Jiahao Xiao, Ye Shen, et al. Large multimodal models evaluation: a survey. *Science China Information Sciences*, 68(12):221301, 2025.

A Localization and Generalization: Additional Detail

A.1 Learning-Free Floors

Table 2 gives the two learning-free predictors against which Sec. 7.2 reads our Dice: a constant mask over the whole page, and the union of every layout component Docling finds. Both are reported on edited pages, on pristine pages and over all pages, because the three numbers measure different abilities: delineating the edited region, refusing to mark anything on an untouched page, and the average of the two that a single corpus-wide figure would report.

Table 2: Localization against predictors that use no learning, on the In-Generation set. Pixel-level Dice, reported separately on edited and pristine pages, because that separation is what distinguishes a detector from a content parser.

Predictor	Edited	Pristine	All pages
Constant mask (whole page)	0.258	0.000	0.129
All layout components (Docling)	0.478	0.009	0.243
Ours	**0.895**	**0.987**	**0.941**

A.2 Region-Level Recall by Component Type

Sec. 7.2 reports that the detector recovers 80.8% of edited regions at 50% coverage and that component area explains most of the spread. Table 3 gives the breakdown. Recall falls almost monotonically as components get smaller, row by row down the table. The clearest exception is captions, which score below footnotes despite being larger, on 22 and 55 regions. Headers and footers are 6.1% of edited regions, and we report them separately because they mark the floor of the method: below roughly one percent of page area, an edit becomes hard to detect, for the reason given in the main text—a running head carries too few words for any statistical judgement about its author, as it would for a human reader.

Table 3: Region-level detection on the In-Generation set, by component type. A region counts as detected when at least 50% of its area is covered by the predicted mask.

Component type	Recall	Mean area	Regions
Table	0.907	15.1%	97
Figure	0.927	15.0%	245
Text	0.858	6.3%	2,056
Caption	0.545	2.6%	22
Footnote	0.600	1.3%	55
Page header	0.031	0.5%	129
Page footer	0.032	0.2%	31
All regions	**0.808**	—	2,635

A.3 What the Cross-Generation Dice Drop Consists Of

A Dice score cannot say what kind of failure it records: a mask placed confidently over the wrong region and a correct mask too faint to survive the threshold both score low. To tell those apart we look at the probability map the model produces before any threshold is applied (Table 4).

The model still knows where the edit is. It has not started pointing at the wrong place. On almost nine Qwen pages in ten, the most confident pixel of the whole page falls inside the edited region—a check that needs no threshold at all—and the probability inside that region remains around twenty times the probability outside, much as on the other two corpora. What changes is not where the model looks but how sure it is: its confidence in the correct region drops by about a quarter, while the background stays put.

Table 4: What the Cross-Generation Dice drop consists of. Measured on edited pages with a non-empty target, on the In-Generation set, the Granite corpus, and a balanced 1,000-page subsample of the Qwen corpus. "Peak of the map inside the target" is threshold-free: it asks whether the single most confident pixel of the page falls within the edited region. The last row replaces the fixed τ=0.5 with the best threshold for each page individually, and so bounds what any recalibration could recover.

	In-Generation	Granite	Qwen
Peak of the map inside the target	0.961	0.972	0.898
Mean probability inside	0.852	0.829	0.644
Mean probability outside	0.037	0.029	0.031
Pixel precision	0.863	0.856	0.745
Pixel recall	0.872	0.847	0.653
Empty masks	0.044	0.039	0.120
Predicted area / true area	1.001	0.996	0.831
Dice at τ=0.5	**0.865**	**0.846**	**0.679**
Dice at the per-page oracle τ	0.872	0.854	0.703

Less confidence produces two failures. The first is silence: on roughly one edited Qwen page in eight the probability never rises above the threshold anywhere, so the mask comes out empty. The second is understatement: where a mask does appear it is correct but too small, covering about four-fifths of the edited region rather than all of it. What the detector marks is right; it marks less of it. The last row of Table 4 shows that no per-page recalibration would recover more than a few points: the missing confidence is not a threshold artefact. This matches what the detection literature reports about generator identity: models differ in how far their statistical signatures sit from human text, and the ones that sit closer are intrinsically harder to detect (Fraser et al., 2025). The narrower that distance, the less evidence any single region carries—still enough to call the page, not always enough to claim the region in full.

B From Pages to Documents

Every number in the main text is a decision about a *page*—the harder measurement, and the only unit on which a mask is defined—while the decision an editor or reviewer actually makes is about a *document*. Aggregating a document's per-page probabilities by their mean gives Table 5: AUROC 0.980 In-Generation and 0.999 Cross-Generation, against 0.944 and 0.929 per page. Per-page errors are largely independent and cancel under an average, so at the unit an operator acts on, an unseen generator costs essentially nothing. These documents average 12 pages with edits spread across them, and a document edited on a single page reduces to the page-level problem; we therefore keep page-level numbers as the headline and report these as what the tool delivers in the setting it was built for.

Table 5: Aggregating page scores to a document decision (mean page probability). Documents, not pages, are the unit an operator acts on, and averaging over a document's pages cancels most of the per-page error.

Test set	Docs	Page AUROC	Doc AUROC	Acc	Prec	Rec
In-Generation	880	0.944	**0.980** [.971,.988]	0.948	0.971	0.942
Cross-Generation	736	0.929	**0.999** [.998,1.000]	0.980	0.940	1.000

C Robustness to Image Laundering

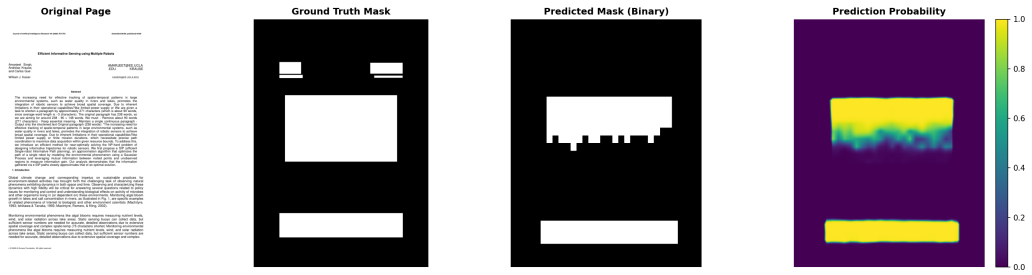
A forensic tool is rarely handed a pristine render: documents circulate as re-compressed attachments, are downsampled by viewers, and are re-exported by intermediate pipelines. Since our detector consumes the page as an image, it might key on high-frequency artifacts any of these would destroy. We test that on a balanced 1,000-page subsample of Cross-Generation against an untouched control on the same pages (Table 6).

Table 6: Robustness under image laundering, on a balanced 1,000-page subsample of the Cross-Generation corpus. The control row is the same model on the same pages with no operation applied, and reproduces the full-corpus Cross-Generation result of Table 1 (AUROC 0.933 against 0.929, Dice 0.714 against 0.699); every other row is read against it.

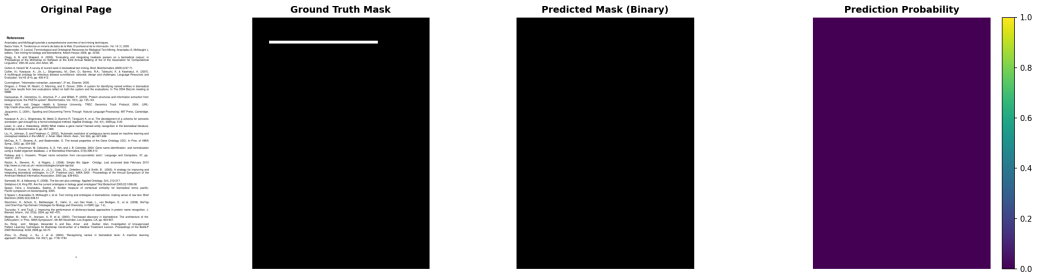
Condition	Acc	AUROC	Spec	Sens	Dice
None (control)	0.854	0.933	0.884	0.824	0.714
JPEG q=90	0.835	0.920	0.828	0.842	0.719
JPEG q=75	0.844	0.928	0.860	0.828	0.707
JPEG q=50	0.839	0.923	0.850	0.828	0.679
Downsample 75%	0.842	0.925	0.888	0.796	0.652
Downsample 50%	0.780	0.850	0.902	0.658	0.490
JPEG 75 + downsample 75%	0.835	0.914	0.862	0.808	0.639
Gaussian blur σ=0.6	0.853	0.926	0.910	0.796	0.661

JPEG re-compression, even at q=50, leaves the page-level decision statistically unchanged (0.920–0.928 AUROC against 0.933) and localization largely intact (0.679 Dice against 0.714): the evidence does not live in compression-fragile high-frequency artifacts, unlike classical image forensics, where JPEG laundering is often fatal. Losing resolution is what hurts. Halving the page drops AUROC to 0.850, concentrated in sensitivity (0.824 $\rightarrow$ 0.658) while specificity rises, with localization degrading earlier still (0.714 $\rightarrow$ 0.490)—the predictable failure mode of a detector reading typographic evidence: once glyph-level detail is gone the smallest edits become invisible and the model withholds a verdict rather than manufacturing false accusations. The envelope is an honest one: documents as they circulate, at or near native page resolution.

D Qualitative Results



(a) A text page. The main edited block is found, but the two small regions at the top are missed and a region the ground truth does not mark is predicted at the bottom.



(b) A references page whose single edited strip is below the size at which the detector responds: the predicted mask is empty and the probability map uniformly near zero. This is the dominant failure mode of Sec. 7.2.

Figure 3: Characteristic failure modes of DocAIdet, chosen over best cases. Each panel shows, left to right, the rendered page, the ground-truth mask, the thresholded prediction and the predicted probability map.

Figure 3 illustrates representative test-set examples, with additional cases in Figure 4 and in our repository[2]. Most detection failures involve minor modifications (short text spans, isolated edits), aligning with established AIGC-detection literature: small-scale inputs contain insufficient generative patterns to reliably distinguish synthetic from authentic content (Fraser et al., 2025; Mitchell et al., 2023; Tian et al., 2024). Since our architecture is optimized to minimize false positives, it rarely flags them—conservative behavior we consider methodologically rigorous, since flagging regions where the statistical distance between human and AI distributions has all but vanished would indicate over-reliance on spurious correlations rather than genuine generative signals. It also confirms empirically that DocAIgen introduces no low-level export artifacts or structural fingerprints: were such trivial cues present, the model would exploit them to detect even the smallest modifications.

Larger modifications are consistently identified, with imperfections limited to boundary imprecision or incomplete coverage. Since our objective is the robust localization of modified regions, these deviations do not compromise forensic utility: the outputs remain actionable as a spatial prior, refinable via standard post-processing if pixel-perfect precision were required.



Figure 4: Further qualitative examples of DocAIdet on test pages. Each row shows the rendered page, the ground-truth mask, the thresholded prediction and the predicted probability map.

E Additional Quantitative Results

E.1 Are the Edits On-Topic?

Sec. 6.1 claims that an edit changes who wrote a component and nothing else about the page, and Sec. 8 reports the consequence for the benchmark. This appendix gives the measurement behind it.

The concern is specific: if the generator replaced a paragraph of a molecular-biology paper with text about quarterly sales, the benchmark would be solvable by noticing a change of subject, and a detector could score well without ever learning anything about machine-written text. DocAIgen is built against this—the generator is conditioned on the component it replaces and on the "similar" similarity level (Appendix G.6)—but the property has to be verified on the corpus that was actually produced.

We measure it as a displacement between two reference points. For each edited page we compute its lexical similarity to the rest of its own document. The ceiling is the same quantity computed for the corresponding *pristine* page: how on-topic an untouched page of that document is. The floor is the page's similarity to a randomly chosen unrelated document. Over $3{,}000$ page triples the ceiling is 0.309 and the floor 0.049; edited pages sit at 0.291, retaining 92.8% of the span between them. Only 0.10% of edited pages—roughly one in a thousand—are closer to a random document than to their own.

Edited pages are therefore almost exactly as on-topic as the pages they replace. Topical incoherence is not a cue available to any detector on this benchmark, ours included, which is what makes the results in Sec. 7 attributable to the generative signature of the content rather than to its subject matter.

E.2 Calibration and False Positives across Domains

Every number in the main text concerns *ranking*: whether edited pages score above pristine ones. Deployment needs something stricter. An operator does not rank a submission stream, they act on individual pages, and to do that they need the score itself to be meaningful—a page marked 0.8 should be edited about eighty percent of the time, or the number cannot be used to trade off missed edits against false accusations. That property is calibration, and it does not follow from good AUROC; a model can rank perfectly while being systematically overconfident.

We measure it as expected calibration error: bin the pages by predicted probability and ask, in each bin, how far the average prediction is from the observed fraction of edited pages. On the Cross-Generation corpus the detector's ECE (15 bins) is 0.073: its probabilities are already within about seven points of the truth with no post-processing, which is unusual for a network trained with focal loss and means it is usable as shipped. Fitting a single temperature parameter ($T = 1.75$) more than halves that, to 0.033, at no retraining cost.

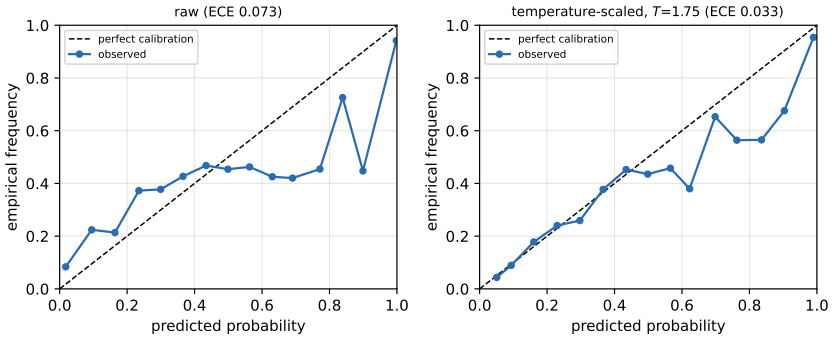


Figure 5: Reliability diagram on the Cross-Generation corpus ($9{,}128$ pages). The detector is already usable out of the box (ECE 0.073); a single temperature parameter brings it to 0.033.

Figure 5 shows the reliability diagram before and after scaling. One thing an aggregate cannot show is whether a detector that behaves well on average is quietly concentrating its false alarms on one

kind of document—a real risk here, since writing conventions differ sharply between disciplines and a tool that suspected every mathematics paper would be unusable whatever its average accuracy. Grouping pristine pages by their document's arXiv primary category, the false-positive rate is flat: 0.081 (physics) to 0.127 (q-bio) around a global 0.118 In-Generation, and 0.018 (q-bio) to 0.138 (physics) Cross-Generation around a global 0.116. The extremes even change places between the two settings, which is what one expects from sampling noise rather than from a systematic bias against a field.

E.3 Paired Significance Tests

A margin between two averages can always be questioned: perhaps one method simply got an easier draw of pages. Because every baseline in this paper was run on exactly the same pages as our detector, that question can be settled directly, by comparing the two methods page by page instead of comparing their averages.

The test is then simple to state. Set aside every page on which the two methods agree—they carry no information about which is better—and count only the pages where they disagree. If the methods were equally good, a disagreement should fall either way with equal probability, so the two counts should be about equal; McNemar's exact test is the probability of seeing a split at least as lopsided as the observed one under that assumption. We report those two counts, that probability, and—for the ranking rather than the decisions—a bootstrap over pages that resamples the same page indices for both methods, so the interval it produces is an interval on the *difference* in AUROC rather than on each method separately (Table 7).

Table 7: Paired comparisons on identical pages. Every difference is significant and no AUROC interval crosses zero. "Only ours" counts pages our detector classifies correctly and the baseline does not; "only base" the reverse.

Test set	Baseline	Only ours / base	McNemar p	ΔAUROC
In-Generation	Text-only	577 / 137	4.7×10^{-65}	$+0.136$ [.119,.152]
In-Generation	SegFormer-B0	582 / 116	1.7×10^{-75}	$+0.194$ [.174,.214]
Cross-Generation	Text-only	757 / 246	4.6×10^{-61}	$+0.077$ [.063,.092]
Cross-Generation	SegFormer-B0	1058 / 202	2.9×10^{-140}	$+0.286$ [.265,.306]
Balanced subset	TruFor	411 / 58	1.4×10^{-66}	$+0.413$ [.373,.452]

The splits are not close. Against SegFormer-B0 on Cross-Generation, $1{,}058$ pages are read correctly by our detector alone and 202 by the baseline alone—better than five to one, where an even split is what equal ability would produce—and every row of the table tells the same story at a p-value below any conventional threshold. No bootstrap interval on the AUROC gap crosses zero. The practical reading is that our margin is not a handful of hard pages tipping an average: where the two methods disagree, ours is the one that is right three to seven times as often.

E.4 Deployment Cost

Screening a document collection is only meaningful if the screening is affordable, and this is where a small specialized model has an argument that accuracy tables do not capture. DocAIdet has 265.4M parameters, of which 37.3M are trained (LoRA adapters and decoder heads). At full page resolution on a single H100 it processes 26.0 pages per second—38 ms per page—with a peak memory footprint of 5.3 GiB, so it fits comfortably on consumer hardware and clears roughly $94{,}000$ pages per hour. The MLLM baselines it outperforms in Table 1 are orders of magnitude larger, are served remotely, and cannot produce a segmentation mask at any cost. For an institution screening a submission stream, that difference is the difference between a deployable tool and a demonstration.

E.5 Full Classification Metrics

Table 1 reports AUROC and Dice, the two axes on which the methods differ. AUROC is a ranking statistic, and a method can rank pages tolerably while being unusable at any fixed threshold, so Table 8 gives the remaining classification metrics at the fixed threshold of 0.5 for every competitor

that reaches a non-trivial ranking. Two protocol details of Table 1 belong here. Our row is scored on the full Cross-Generation corpus (9,128 pages) and every baseline on the balanced 3,000-page subsample they were all run on; restricting our detector to those pages changes its AUROC by less than 0.001, and the paired tests of Appendix E.3 compare on identical pages in any case. RIGID needs the whole page as a reference frame and so has no figure-crop row.

Table 8: Classification detail at threshold 0.5. Specificity is the column that separates a usable detector from an unusable one: several piecewise combinations and the text detector fine-tuned on whole pages reach their rankings by accusing most genuine pages, which is invisible in an AUROC column.

Test set	Model	F1	Acc	Prec	Rec	Spec
In-Generation	Piecewise, page-level (off-the-shelf)	0.350	0.533	0.575	0.251	0.815
	Piecewise, region-level (off-the-shelf)	0.640	0.565	0.546	0.774	0.356
	SegFormer-B0	0.705	0.647	0.606	0.842	0.453
	AI-text detector (fine-tuned)	0.729	0.659	0.605	0.918	0.401
	AI-text detector (fine-tuned on blocks)	0.740	0.785	0.940	0.610	0.961
	Piecewise, page-level (trained)	0.706	0.594	0.553	0.976	0.213
	Ours	**0.857**	**0.860**	0.877	0.838	0.882
Cross-Generation	Piecewise, page-level (off-the-shelf)	0.222	0.504	0.513	0.141	0.866
	Piecewise, region-level (off-the-shelf)	0.638	0.531	0.520	0.826	0.237
	SegFormer-B0	0.639	0.572	0.552	0.759	0.385
	AI-text detector (fine-tuned)	0.748	0.687	0.626	0.931	0.443
	AI-text detector (fine-tuned on blocks)	0.814	0.831	0.906	0.739	0.923
	Piecewise, page-level (trained)	0.706	0.590	0.551	0.982	0.197
	Ours	**0.856**	**0.860**	0.878	0.835	0.884

E.6 Sensitivity–Specificity Trade-off and Threshold Analysis

The threshold is fixed at 0.5 on every test set and never tuned, and the model is deliberately biased toward specificity: in AIGC detection a false positive is an unjust accusation. The results in Table 1 should be read in that light, since such high specificity inevitably costs sensitivity. Table 9 shows how the metrics shift simply by moving the decision threshold at inference time. Should a use case require a more balanced operating point, it can also be shifted during training, by tuning the Focal Loss α (Eq. 6) or adding class weights to the Dice Loss (Eq. 7).

Table 9: Page-level classification across the decision threshold. $\star$ marks the fixed operating point used everywhere in the paper. Raising the threshold buys specificity cheaply up to ≈ 0.75—In-Generation specificity reaches 0.988 at a cost of 10 points of sensitivity—after which it saturates. These are *page-level* decisions; the pixel-level segmentation metrics behave differently and are reported in Sec. 7.2.

	In-Generation				Cross-Generation			
Threshold	Sens.	Spec.	Prec.	F1	Sens.	Spec.	Prec.	F1
0.10	0.970	0.547	0.682	0.801	0.928	0.722	0.769	0.841
0.25	0.934	0.702	0.758	0.837	0.898	0.802	0.819	0.857
0.50 $\star$	0.838	0.882	0.877	0.857	0.835	0.884	0.878	0.856
0.75	0.738	0.988	0.984	0.843	0.795	0.936	0.926	0.855
0.90	0.732	0.993	0.990	0.841	0.780	0.947	0.937	0.851
0.95	0.732	0.993	0.990	0.841	0.773	0.953	0.943	0.850
0.99	0.721	0.994	0.991	0.834	0.753	0.964	0.954	0.842

F DocAIDet: Training and Architecture Details

F.1 Loss Functions

Phase 1 (segmentation) uses a combined focal and Dice loss following the SAM framework,

$$\mathcal{L}_{\text{total}} = 2.0\,\mathcal{L}_{\text{focal}} + 1.0\,\mathcal{L}_{\text{Dice}}, \tag{5}$$

where the focal loss is

$$\mathcal{L}_{\text{focal}} = \alpha(1-p_t)^{\gamma}\,\text{BCE}(z,y) \tag{6}$$

($\alpha = 0.5$, $\gamma = 2.0$), BCE denotes binary cross-entropy with logits, z are the raw model predictions, $y \in \{0,1\}$ are the ground truth binary masks, and p_t is the predicted probability of the true class; and the Dice loss measures prediction–ground truth overlap ($\epsilon = 0.01$):

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2\sum p_i y_i + \epsilon}{\sum p_i + \sum y_i + \epsilon}. \tag{7}$$

Phase 2 (classification) uses weighted cross-entropy with weights $w_1 = 0.66$, $w_2 = 0.33$:

$$\mathcal{L}_{\text{CE}} = -(w_1 y_1 \log(p_1) + w_2 y_2 \log(p_2)). \tag{8}$$

F.2 Training Protocol and Hyper-parameters

Figure 6 presents the training curves for Phase 1, displaying loss curves (left) and validation metrics (right). Figure 7 shows the training evolution of Phase 2, illustrating the progression of both loss values and performance metrics throughout the training process.

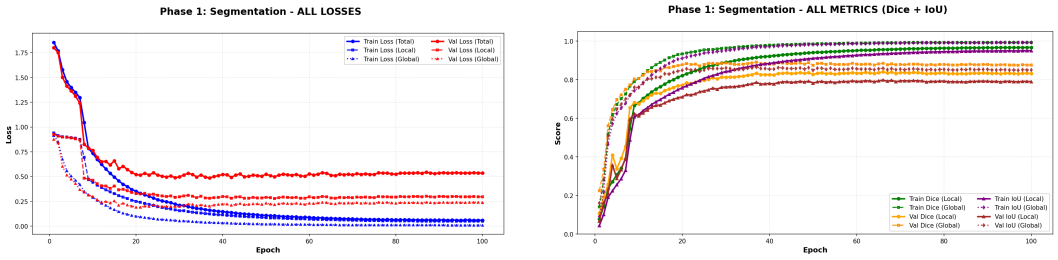


Figure 6: Phase 1 training curves: loss progression (left) and validation metrics (right).

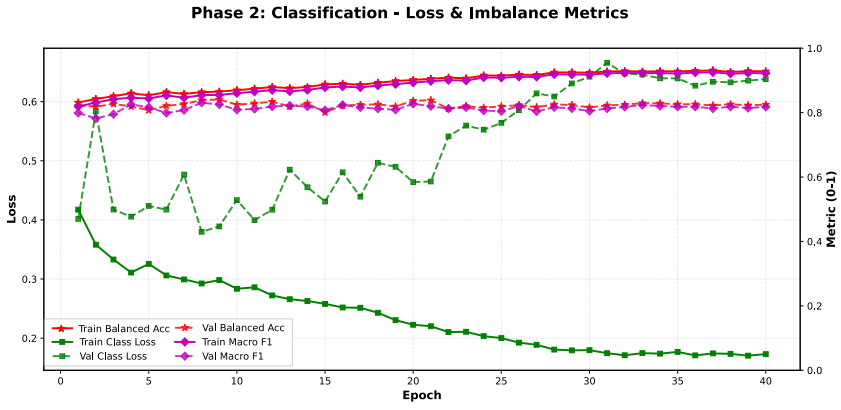


Figure 7: Phase 2 training curves showing loss and metrics evolution.

Phase 1: LoRA configuration. LoRA is applied with $r = 16$, $\alpha_{\text{LoRA}} = 32$, dropout 0.1, no bias, in feature-extraction mode (Table 10), on the query, key, value and output projections and on the gate, up and down projections of the backbone.

Table 10: LoRA configuration parameters.

Param	Value
`use_lora`	true
`r`	16
`lora_alpha`	32
`lora_dropout`	0.1
`bias`	"none"
`task_type`	"Feature Extraction"
`target_modules`	`q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj`

Phase 1: optimizer and scheduler. Training is conducted for 40 epochs, and the checkpoint corresponding to the epoch with the best performance on the validation set is selected. Phase 1 uses AdamW with an initial learning rate of 1.0×10^{-4} and weight decay 0.05 (Table 11). The learning rate is controlled through a sequential scheduler that combines a warmup phase and a cooling phase: LinearLR during the first 10% of training steps, scaling from 10^{-9} to the initial learning rate 10^{-4}, followed by CosineAnnealingLR with $\eta_{\min} = 10^{-6}$ for the remaining steps. This approach improves training stability during initial iterations and enables smooth convergence toward minimal learning rate values.

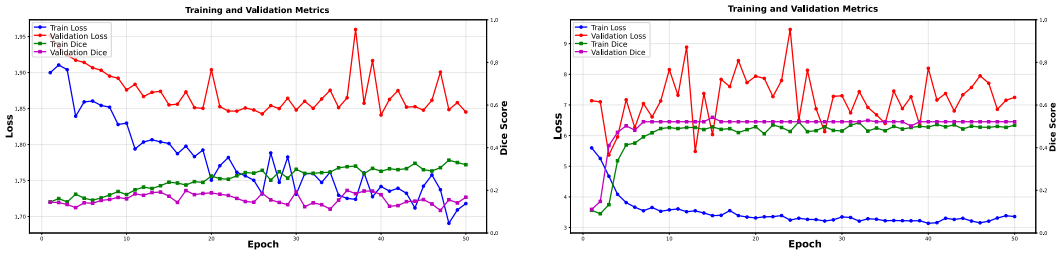
Table 11: Phase 1 optimizer configuration.

Param	Value
Optimizer	AdamW
Initial learning rate	1.0×10^{-4}
Weight decay	0.05
Warmup	LinearLR, first 10% of steps
Main scheduler	CosineAnnealingLR, $\eta_{\min} = 10^{-6}$

Phase 2: hyper-parameters. Phase 2 uses the same optimizer (AdamW) with a higher learning rate of 1.0×10^{-3}, weight decay 0.05 and no scheduler. As discussed in the main text, 1–2 epochs are sufficient for this classification head to align with the rich segmentation embeddings it builds upon; we nevertheless trained for 40 epochs to demonstrate that training beyond 1–2 epochs is unnecessary.

F.3 Single-Stream Training Curves

Figure 8 shows the training dynamics behind Sec. 8.1: one decoder trained on the visual pyramid alone and one on the language decoder's hidden states alone, with no cross-attention and no fusion of any kind, from scratch on a common preliminary subset for 50 epochs under the same recipe.



(a) Standalone vision encoder

(b) Standalone LLM decoder

Figure 8: Single-stream training dynamics. (a) The visual pyramid alone *cannot fit the task*: training Dice stalls near 0.3 and validation Dice oscillates around 0.15 with no trend—not overfitting but a failure to learn, the model being denied the signal rather than generalization. (b) The semantic states alone *learn and then stop*: both curves rise steeply for six epochs, after which validation Dice is flat at ~ 0.5 for the remaining 44 while validation loss drifts upward. That ceiling is structural rather than a matter of budget, since a sequence of hidden states carries no spatial index with which to resolve *where* on the page the evidence sits.

F.4 Backbone Swap: Full Metrics

Table 12 reports the full metric set behind Sec. 8.2; the paired tests are given in that section.

Table 12: Backbone swap at the deployed model's own training budget (Sec. 8.2). Only the pretrained weights differ; corpus, schedule, batch size, learning rates and seed are identical, with no warm start. "Val. Dice" is the best phase-1 validation score; the remaining columns are measured on the held-out In-Generation test set. Brackets give 95% bootstrap confidence intervals.

Backbone	Pretraining	Val. Dice	Acc	Macro-F1	AUROC	Dice
SmolVLM-256M-Instruct	general-purpose VLM	0.831	0.800	0.799	0.905 [.892,.917]	0.714
SmolDocling-256M	document conversion	0.881	**0.860**	**0.860**	**0.944** [.935,.953]	**0.895**

F.5 Architecture Details

Sec. 5 specifies the model as a set of equations. This appendix gives what those equations leave implicit: the concrete shapes each stage produces, and where the trainable parameters sit. Table 13 lists them for a page processed at 1190×1683, which yields a 4×4 tiling plus one global view ($N+1 = 17$ encoder inputs).

Table 13: Stage-by-stage specification. Shapes omit the batch dimension. "Trained" counts parameters updated during our training, in millions; the backbone contributes only through its LoRA adapters. Six fusion blocks are applied, three to the recombined local pyramid and three to the global context map.

Stage	Output shape	Trained (M)
Vision encoder (layers 3, 7, 11)	$3 \times (17 \times 768 \times h \times w)$	5.77 (LoRA)
Language decoder hidden states	$L \times 576$	shared LoRA
Recombiner (Eq. 1)	$3 \times (768 \times 4h \times 4w)$	0
Semantic projection $\mathbf{W}_s$, $\times 6$ (Eq. 2)	$L \times 768$	2.66
Cross-modal attention, 8 heads, $\times 6$	$HW \times 768$	14.17
Gate MLP 768→192→1, $\times 6$ (Eq. 3)	$HW \times 1$	0.89
Segmentation head (proj → refine → FFN → 1×1)	$1 \times H_0 \times W_0$	11.04
Classification head (Eq. 4 → fuse → linear)	1	2.76
Total		**37.3**M / 265.4M

Two design points are worth expanding beyond the equations.

Why the recombiner is parameter-free. The tiled encoder returns each tile independently, so a segmentation head reading them in sequence would have to re-learn the page geometry from scratch—a mapping that is known exactly and constant. Eq. 1 therefore restores it by construction. Figure 9 shows the rearrangement: local tiles are re-tiled into a single map in acquisition order, while the global view is kept aside as a page-level context stream, so that both a fine-grained and a whole-page representation reach the heads. Making this step learnable was tried and offered no benefit, which is the expected outcome for an operation with a closed form.

Why the fusion is gated rather than additive. A document page is mostly whitespace and unedited content. An ungated residual would force every spatial position to absorb semantic context, including positions with no text beneath them, which in early experiments blurred mask boundaries. The gate of Eq. 3 is computed from the visual features alone, so the model decides where semantics are relevant before consulting them; empirically the gate saturates low over margins and figures and high over body text.

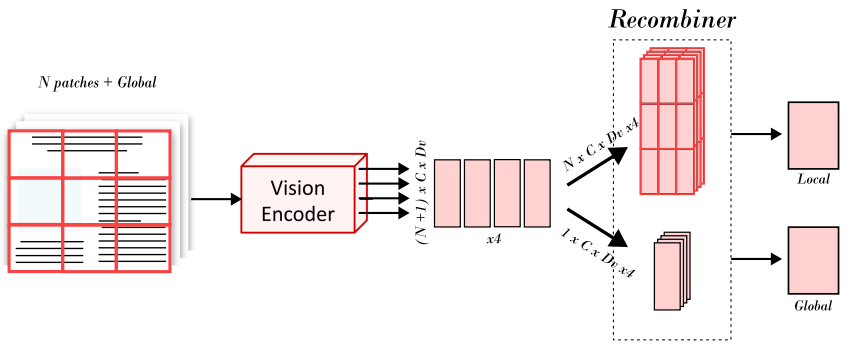


Figure 9: Internal architecture of the Recombiner module: per-tile encoder features are reassembled into a single spatially coherent map, with the global view retained as a separate page-level context stream.

G DocAIgen: Details

G.1 Database Schema

The following shows an example of what is stored in the database automatically generated by DocAIgen.

- **pdf_path**: Full path to the original PDF file (string).
- **is_modified**: Boolean indicating whether the document has been artificially modified (`true`/`false`).
- **modifications**: Array of objects specifying the modified regions when `is_modified = true`. Each modification includes:
  - **component_type**: Type of affected component (`"section_header"`, `"footnote"`, etc.).
  - **type**: Semantic subtype (`"text"`, `"image"`).
  - **page**: Affected page number (int).
  - **bbox**: Bounding box of the modified region, with structure:
    - **page**, **x**, **y**: Coordinates and page.
    - **width**, **height**: Bounding box dimensions.
    - **coord_origin**: Coordinate system origin (e.g., `"CoordOrigin.BOTTOMLEFT"`).
    - **raw**: Original layout coordinates (`left`, `top`, `right`, `bottom`).
- **component_inventory**: Complete structural inventory of the PDF, containing:
  - **pages**: Dictionary indexed by page number (`"1"`, `"2"`, ...), where each page contains:
    - **page_number**: Page number (int).
    - **text_sections**: Array of text blocks, each with:
      - **type**: `"TextItem"`, `"SectionHeaderItem"`, etc.
      - **label**: Semantic label (`"text"`, `"section_header"`, `"page_header"`, etc.).
      - **parent_type**: Parent container (`"body"`, `"picture"`, `"table"`).
      - **preview**: Representative text snippet.
      - **length**: Character length.
      - **bounding_box**: Same structure as `modifications.bbox`.
      - **ref**: Internal reference (`"#/texts/35"`).
    - **tables_in_page**: Array of detected tables.
    - **images_in_page**: Array of detected images.

G.2 Use Cases and Extensibility

DocAIgen's modular architecture is explicitly designed for maximum extensibility. Below, we present several practical use cases that our framework *already* supports natively by simply selecting and combining its constituent modules. While these functionalities are highly valuable and could constitute standalone tools, we provide them here fully open-sourced:

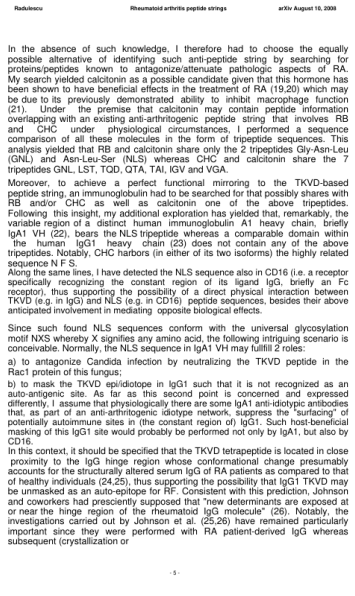
- **Format Conversion:** Integrating the *Docling* and *Exporter* modules enables multi-format document conversion (e.g., transforming a photographed physical document into a structured PDF). This functionality is readily extensible to support any desired output format.
- **Data Ingestion:** *Docling* and *Database* modules enable the automated analysis of document collections, constructing structured SQL databases with metadata, raw text and images.
- **Dataset Curation:** By combining the *Docling*, *AI Edition*, and *Database* modules, users can extract document components and process them with generative models to produce specialized AI-generated image or text datasets.

- **Synthetic Generation:** The full pipeline can be deployed for controlled document generation tailored to various downstream tasks (e.g., training Document Question Answering (DocQA) models or Spatial Entity Localization).

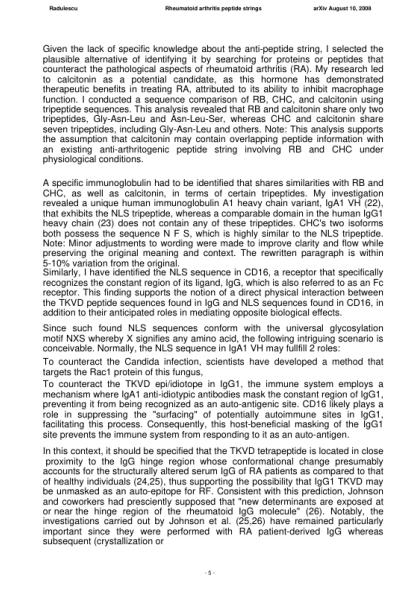
Furthermore, DocAIgen is designed for high usability, utilizing a dedicated configuration file that allows users—regardless of programming expertise—to operate and fine-tune the system without requiring modifications to the source code[1].

G.3 Qualitative Comparison

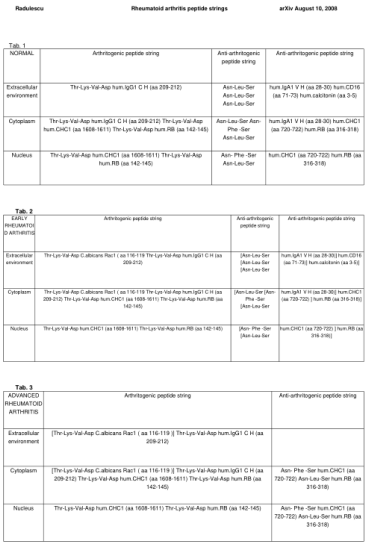
Figure 10 presents three examples illustrating the functionality of DocAIgen, each demonstrating a different modification type. As observed, the sole difference between the real and modified samples is the content itself. This ensures the model is forced to detect intrinsic generative patterns rather than relying on visual artifacts. Additional examples are available in our repository[1].



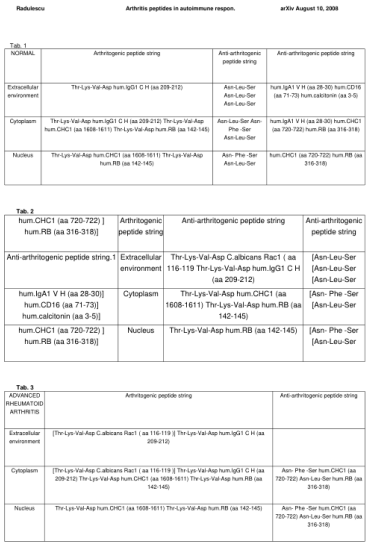
(a) Original page



(b) Original page with text modifications

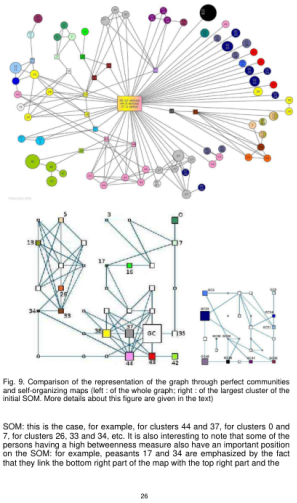


(c) Original page

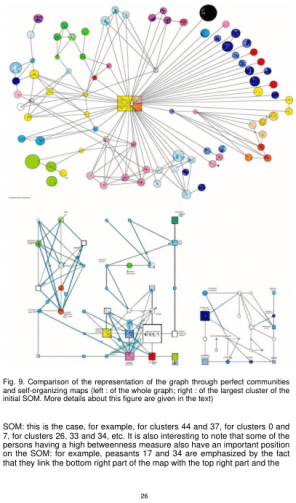


(d) Original page with table modifications

Figure 10: DocAIgen examples: a text edit (top row) and a table edit (bottom row). Each row displays the original page (left) and its AI-modified version (right).



(a) Original page



(b) Original page with image modifications

Figure 11: DocAIgen example: a figure edit. Original page (left) and its AI-modified version (right).

G.4 Quantitative Analysis

To complement the qualitative evaluation of our framework, we conduct a quantitative analysis of the structural, visual, and statistical fidelity of the AI-generated (AIG) documents compared to their real counterparts, through four distinct forensic tests:

- **Frequency Analysis (FFT):** We evaluate the Fast Fourier Transform (FFT) spectra to identify synthetic frequency fingerprints. Generative models often leave characteristic traces in high-frequency bands; thus, we calculate total energy and the distribution ratios across low, mid, and high frequencies.
- **OCR Confidence and Lexical Integrity:** To evaluate character-level distortions, legibility, and text degradation, we process the documents through a standard OCR engine. We compare mean confidence scores, total word counts, Out-Of-Vocabulary (OOV) rates, and gibberish ratios.
- **Layout Geometry:** Synthetic rendering can sometimes produce "hyper-regular" alignments or abnormal spacing. We measure the standard deviation of left and right margins, as well as the Coefficient of Variation (CV) of word gaps.
- **Error Level Analysis (ELA):** To isolate non-uniform compression signatures typical of manipulated images, we apply ELA, analyzing the mean and standard deviation of compression errors across text regions and background areas.

Results and discussion. We tested $2{,}500$ AIG pages generated by DocAIgen against $2{,}500$ pristine real pages. The comparative results are presented in Table 14. The numerical differences across all four forensic domains are negligible. The minor fluctuations present are purely stochastic, arising from the natural variability inherent to distinct document layouts and contents, rather than their source (real vs. AIG). This extremely high statistical overlap demonstrates that DocAIgen successfully replicates the physical and digital properties of real-world documents. Consequently, this validates our methodological claim: detection models trained on this benchmark are forced to learn meaningful, high-level generative patterns, as low-level structural, frequency, or compression shortcuts are absent. This behavior is also evident in the qualitative results of Appendix D and in the shortcut audit of Sec. 8.

Table 14: Quantitative evaluation of low-level artifacts between AIG and real documents across $5{,}000$ samples. Metrics indicate near-perfect statistical alignment across frequency, legibility, geometry, and compression domains.

Metric	Fake (AIG)	Real
Frequency Domain (FFT)		
Total energy	$(2.19 \pm 0.11) \times 10^{11}$	$(2.19 \pm 0.11) \times 10^{11}$
Low ratio	0.980 ± 0.009	0.980 ± 0.009
Mid ratio	0.007 ± 0.004	0.007 ± 0.004
High ratio	0.013 ± 0.006	0.013 ± 0.006
OCR Integrity & Confidence		
Mean conf.	49.43 ± 16.64	49.90 ± 16.41
Word count	170.6 ± 84.7	170.5 ± 83.8
OOV rate	0.486 ± 0.192	0.466 ± 0.190
Gibberish rate	0.023 ± 0.024	0.024 ± 0.023
Layout Geometry		
L-margin std	123.5 ± 18.3	123.6 ± 18.4
R-margin std	125.3 ± 17.7	125.3 ± 18.2
Word gap CV	1.386 ± 0.299	1.352 ± 0.290
Error Level Analysis (ELA)		
Mean	0.735 ± 0.261	0.728 ± 0.268
Std	1.749 ± 0.425	1.743 ± 0.431
Text mean	3.123 ± 0.737	3.108 ± 0.690
Bg. mean	0.541 ± 0.191	0.536 ± 0.195

G.5 DOCLING STRUCTURED VARIABLE

Docling processes PDF documents and returns a `ConversionResult` object containing all structured information about the document. Below we describe its hierarchical structure.

Level 1: ConversionResult. The root structure contains global metadata and data collections:

```
ConversionResult(
  input: InputDocument,            # Input PDF information
  document_hash: str,              # Unique document hash
  status: ConversionStatus,        # SUCCESS | FAILURE | ...
  errors: List[str],               # Errors during conversion
  pages: List[Page],               # Processed pages
  assembled: AssembledUnits        # Unified elements
)
```

Level 2: InputDocument. Contains metadata about the input file:

```
InputDocument(
  file: PurePosixPath,             # PDF file path
  format: InputFormat,             # PDF, DOCX, etc.
  filesize: int,                   # Size in bytes
  pagecount: int,                  # Total pages (23)
  document_hash: str,              # Document hash
  valid: bool                      # Document validity
)
```

Level 3: Page. Each page contains size information and layout predictions:

```
Page(
  pageno: int,                     # Page number (0-indexed)
  size: Size,                      # width=612.0, height=792.0 (points)
  predictions: PagePredictions     # Element predictions
)
```

Level 4: PagePredictions (LayoutPrediction). Contains clusters of detected elements on the page:

```
PagePredictions(
  layout: LayoutPrediction(
    clusters: List[Cluster],       # Elements: texts, headers, images
    tablemap: dict,                # Table mapping (if exists)
    figures: dict                  # Figure detection (None)
  )
)
```

Level 5: Cluster (content element). Each cluster represents an element (paragraph, heading, image, etc.):

```
Cluster(
  id: str,                         # Unique ID (e.g., "13", "5")
  label: DocItemLabel,             # TEXT | SECTIONHEADER | PICTURE | ...
  bbox: BoundingBox,               # Element bounding box
  confidence: float,               # Confidence: 0.51-0.99
  cells: List[PdfTextCell],        # Text cells within cluster
  children: List[Cluster]          # Nested clusters
)
```

Level 6: BoundingBox. Defines element coordinates on the page:

```
BoundingBox(
  l: float,                        # left (77.4)
  t: float,                        # top (38.687)
  r: float,                        # right (277.637)
  b: float,                        # bottom (93.063)
  coord_origin: CoordOrigin        # TOPLEFT
)
```

Level 7: PdfTextCell. Represents each text cell within a cluster:

```
PdfTextCell(
  index: int,                      # Index in cluster (0, 1, 2, ...)
  text: str,                       # "Transactions of the SDPS"
  orig: str,                       # Original text
  confidence: float,               # 1.0 (very confident)
  textdirection: TextDirection,    # LEFTTORIGHT
  font_key: str,                   # "TT0" (font reference)
  font_name: str,                  # "TimesNewRomanPSMT"
  rendering_mode: RenderingMode,   # UNKNOWN, VISIBLE, etc.
  rect: BoundingRectangle,         # Specific cell coordinates
  rgba_color: RGBAColor            # r=0, g=0, b=0, a=255
)
```

Level 8: AssembledUnits (unified elements). Collects all processed elements in a unified structure; the AI edition module operates on the `text` field of these elements and on the image payload of `ImageElements`, and on nothing else:

```
AssembledUnits(
  elements: List[Union[TextElement, ImageElement]]
)

TextElement(
  label: DocItemLabel,             # Element type
  id: str,                         # Unique ID
  pageno: int,                     # Page number (0-indexed)
  cluster: Cluster,                # Cluster reference
  text: str                        # Combined text
)
```

G.6 Generation Hyperparameters

This appendix details all hyperparameters used in DocAIgen for document generation. The main generation aspects are discussed in Sec. 6.1; here we provide complete technical specifications.

G.6.1 Generative Models

DocAIgen supports an extensible pool of generators, categorized into Language and Vision domains. The *Language* models include DialoGPT (Zhang et al., 2020), Phi-3-mini-4k-instruct (Abdin et al., 2024), Qwen2.5 (Yang et al., 2024), Qwen3 (Yang et al., 2025), Llama-3.2 (Grattafiori et al., 2024), and Granite (Granite Team, 2024), while the *Vision* models consist of Stable-Diffusion-v1.5 (Rombach et al., 2022) and SD-Turbo (Sauer et al., 2024). Notably, our framework affords the flexibility to selectively assign specific vision or language models to distinct document components as required (see Sec. 4).

The benchmark reported in this paper instantiates that pool as follows. The **In-Generation** corpus (training, validation, and test splits) edits textual and tabular components with Llama-3.2-3B-Instruct and pictorial components with SD-Turbo, over a pool of 398 source documents. The **Cross-Generation** corpora are built from a *disjoint* pool of 503 source documents—the two pools

share no arXiv identifier, so cross-generation evaluation varies the generator *and* the documents simultaneously—and are edited by two language models never seen during training: Qwen3-4B-Instruct-2507 and Granite-4.0-h-small.

G.6.2 SIMILARITY LEVEL

Users control how closely the synthetic content of the components resembles the original through four predefined levels: very similar, similar, moderately different, and totally different. While the similarity parameter is integrated into DocAIgen to maximize its versatility for diverse downstream applications, we deliberately fix it to the "similar" setting for training DocAIdet. We consider this the optimal configuration because it generates synthetic content that aligns closely with the original context, preventing trivial semantic discrepancies. Consequently, the model is forced to learn deep, intrinsic AI-generative patterns rather than potentially overfitting to thematic or contextual inconsistencies. Conversely, under the "very similar" regime, the generative forensic signals are diluted, unnecessarily complicating the training process without contributing to robust AIGC identification. Therefore, the "similar" level provides the ideal balance for isolating the generative footprint during training.

G.6.3 MODIFICATION RATIO

This parameter governs the selection of document components targeted for modification. Initially, the framework parses the document to extract all underlying components—such as titles, paragraphs, lists, images, tables, footnotes, or formulas—structuring them hierarchically at both the global and page levels. Subsequently, target components can be selected either interactively or automatically via a stochastic ratio. Manual selection affords the user granular control to perform highly specific, targeted edits on individual documents. Conversely, for large-scale automated dataset generation, the stochastic ratio approach is preferred. This ratio defines the proportion of altered content: for instance, a ratio of 0.0 applies no changes, 0.5 alters half of the components, and 1.0 completely regenerates the document. To optimize the dataset for training the segmentation model, a ratio of 0.5 is highly recommended. It modifies half of the components at the document level—that means some pages might have just one element modified while others might have all elements changed. We designed it this way so our DocAIdet learns across all types of contexts.

G.6.4 TEXT GENERATION CONFIGURATION

Temperature and sampling parameters are controlled by the similarity level (Table 15). Additional text generation constraints: minimum tokens 5; maximum tokens 1024; length calculation buffer 0.10 (10% margin); LLM adjustment deviation threshold 0.20 (20%); maximum length adjustment iterations 2.

Table 15: Temperature and sampling parameters per similarity level for text generation.

Similarity Level	Temperature	Top-p	Top-k	Rep. Penalty
Very Similar	0.5	0.95	80	1.0
Similar	**0.7**	**0.90**	**60**	**1.02**
Moderately Different	0.9	0.85	50	1.05
Totally Different	1.2	0.80	40	1.1

G.6.5 TABLE GENERATION CONFIGURATION

Tables are generated with fixed sampling parameters: sampling enabled, temperature 0.3, maximum 3000 new tokens.

G.6.6 IMAGE GENERATION CONFIGURATION

Image generation parameters vary by similarity level (Table 16). There are three image generation modes: image-to-image (generates from the replaced figure), text-to-image (generates from the figure caption), or baseline (generates a random image if the previous modes fail).

Table 16: Image generation parameters (image-to-image) per similarity level.

Similarity Level	Strength	Guidance Scale	Inference Steps
Very Similar	0.25	6.0	10
Similar	**0.45**	**7.5**	**10**
Moderately Different	0.65	9.0	15
Totally Different	0.85	12.0	15

G.6.7 Document Processing and Hardware Configuration

Image resolution scale 8.0; page images and figure/picture images are generated; export tolerance profiles: PDF Strict 0.10 (10%), PDF Basic 0.20 (20%). Generation ran on a single GPU with 24 GB of memory (CUDA, with CPU fallback if unavailable).