Journal of Innovative Image Processing (ISSN: 2582-4252)
www.irojournals.com/iroiip/

LOCUS: Verifying OCR by Local Optimality over Counterfactual String Graphs

Shikhar Dubey[1*], Sanhita Saxena[2]

[1]Intelligent Systems, School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, Himachal Pradesh, India.
[2]School of Mathematical and Statistical Sciences, Indian Institute of Technology Mandi, Himachal Pradesh, India.

E-mail: [1*]s24130@students.iitmandi.ac.in, [2]s25008@students.iitmandi.ac.in

Orcid ID: [1*]0009-0001-9035-4392, [2]0009-0001-1802-2852

Abstract

Optical character recognition engines always return a transcription, so every pipeline must decide, word by word, whether to trust it. Existing verifiers judge the claim in isolation, by engine confidence, by agreement with a second recogniser, or by asking a large vision-language model. We instead ask whether the claim is the best explanation of the image among the strings it could have been confused with. LOCUS constructs a small graph around each claim, comprising plausible single-character edits and independent readings. It scores each candidate string according to how well it explains the real image relative to a clean rendering of that string, thereby cancelling the recogniser's language prior, and uses a relation-aware graph neural network to determine whether the claim is locally optimal. The network is pretrained without human labels from engine agreement and synthetic corruptions and adapts from a few labelled words. We introduce a benchmark of real recognition errors from two engines on five public datasets covering forms, receipts, handwriting and scene text. With 25 labelled words per class, LOCUS ranks correct above incorrect transcriptions better than engine confidence, learned confidence, engine voting, Florence-2 re-recognition and a seven-billion-parameter vision-language judge on every dataset, and attains the best average macro-F1. It is significantly better than re-recognition with a strong recogniser where that recogniser is fallible, comparable where it is in-domain, and behind it on its own scene-text benchmark. Controlled analyses show that the gain comes from the explicit counterfactual evidence and label-free pretraining: simpler learners on the same evidence come within 0.015 macro-F1 of the graph network, and message passing is not needed. LOCUS also rejects most errors whose true string lies outside its graph. A cascade that runs LOCUS only on words a fast recogniser does not confidently confirm keeps its macro-F1 at 1.3–2.6 times the throughput.

Keywords: OCR Verification, Counterfactual Reasoning, Graph Neural Networks, Few-shot Learning, Render Normalisation, Document Analysis.

*** Corresponding Author**

Journal of Innovative Image Processing, December 2026, Volume 8, Issue 4, Pages 1620-1644
DOI: https://doi.org/10.36548/jiip.2026.4.014
Received: 18.09.2026, received in revised form: 21.10.2026, accepted: 13.11.2026, published: 24.11.2026
© 2026 Inventive Research Organization. This is an open access article under the Creative Commons Attribution-NonCommercial International (CC BY-NC 4.0) License

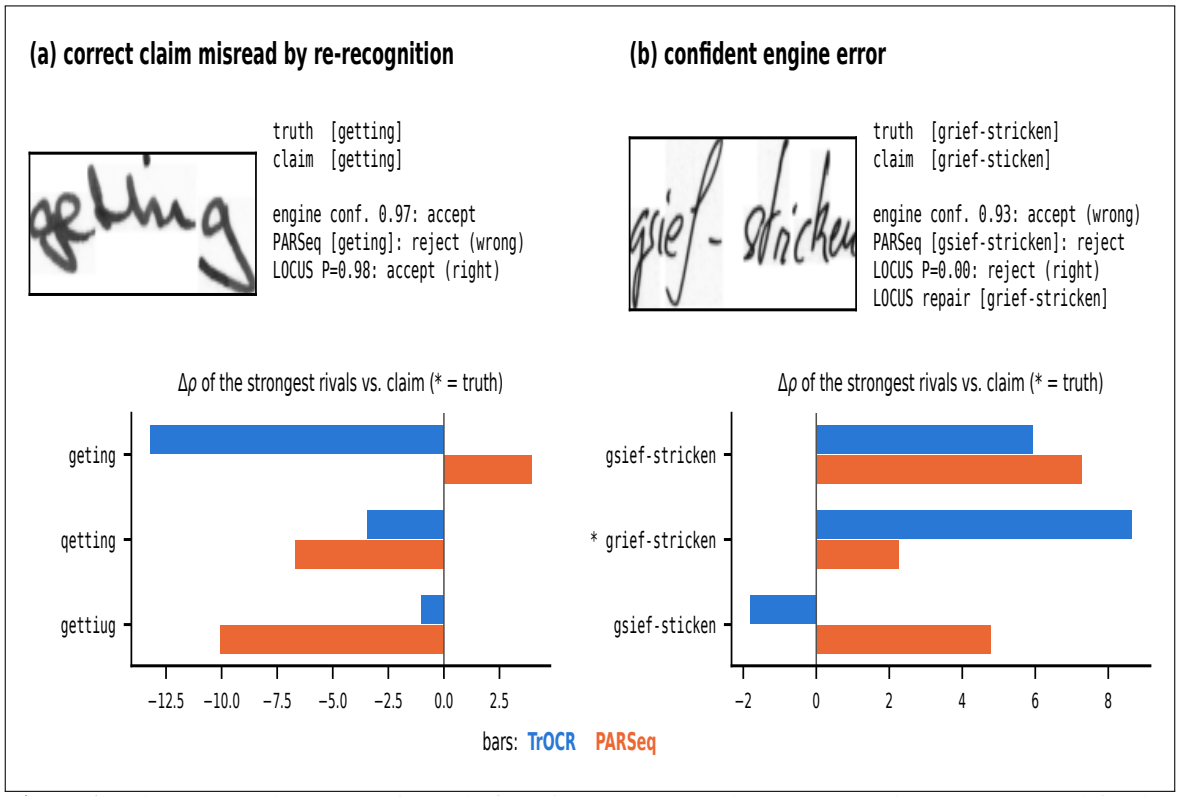

Figure 1. Why compare hypotheses instead of scoring one. Two real IAM test words. (a) A correct claim that PARSeq misreads, so re-recognition rejects it; PARSeq prefers its own reading (orange bar) but TrOCR strongly disfavours it (blue bar), and LOCUS, which weighs both verifiers' evidence for every rival, accepts the claim (this happens for 312 of the 419 correct IAM claims that PARSeq misreads). (b) A confident engine error that engine confidence accepts; LOCUS rejects it and proposes the correct string. Bars: render-normalised evidence of the strongest rivals relative to the claim ($*$: ground truth).

1. Introduction

Optical character recognition (OCR) engines always return a transcription, and modern end-to-end and vision–language readers return fluent text even when they are wrong [1, 2]. Every consumer of OCR (key-information extraction from receipts and forms, archival search, accessibility, data entry) must therefore decide, word by word, whether a transcription can be trusted or must be routed to a human. This is *OCR verification*: given a word image x and a claim h produced by an arbitrary, possibly black-box engine, predict whether h is exactly what is written in x. The three families of verifiers used in practice each have a characteristic blind spot (Figure 1). Engine confidence is unavailable for many engines and overconfident on fluent, plausible errors. Re-recognition and voting (accept h iff a second recogniser reads the same string [3, 4, 5]) is a binary test that inherits every bias of the second recogniser. Vision–language judges [6] contain billions of parameters, cannot be effectively adapted using only a handful of labelled examples and, as we show, perform poorly on the character-level judgments required for OCR verification.

1.1 Key Idea

The right question is not "does the image look like h?" but "is h the best explanation of the image among the strings it could have been confused with?" An OCR error is a near miss: a confusable substitution (`O`/`0`, `l`/`1`), a dropped or inserted glyph, or a case flip. We build around

every claim a small *counterfactual string graph* of its OCR-plausible edits and test whether the claim is a *local optimum* of visual evidence on it. Two problems must be solved. First, the score must be *visual*: a recogniser's likelihood $s(v \mid x)$ mixes how well v explains the pixels with how plausible v is as a string, so rare but correct strings are penalised. We cancel this prior with a *render-normalised likelihood ratio* $\rho(v) = s(v \mid x) - s(v \mid R(v))$, where $R(v)$ is a clean rendering of v. Second, the comparison must be learned from very few labels: a relation-aware graph network with an explicit local-optimality readout is pretrained *without human labels* on images where two engines agree, paired with synthetic corruptions whose repair is known.

1.2 Contributions

1. A formulation of OCR verification as testing the local optimality of a claim among OCR-plausible counterfactuals, which also yields a repair proposal (Section 3).
2. Render-normalised counterfactual evidence that removes the recogniser's string prior, computed with two frozen, architecturally different recognisers. This evidence is the principal contribution: our analyses show that it, together with label-free pretraining, accounts for the gain.
3. LOCUS, a relation-aware GNN over counterfactual string graphs, pretrained label-free and adapted from $k \in \{5, \ldots, 100\}$ labelled words per class. The graph network is a convenient, permutation-invariant learner for this evidence that supports label-free pretraining with a repair objective; we show explicitly that it is not necessary, since simpler learners on the same evidence perform almost as well (Sections 5.3 and 6.3).
4. A verification benchmark of *real* OCR errors on five public datasets (forms, camera and scanned receipts, handwriting, scene text) with two engines, and a comparison with engine and learned confidence, voting, three re-recognition baselines and a 7B vision–language judge (Sections 4–5).
5. An honest analysis: review cost, transfer and recalibration, repair, document-level significance with a design-effect correction, recogniser dependence including correlated recogniser failures, graph coverage, graph capacity, accuracy–throughput trade-offs, pseudo-label noise, rendering sensitivity, message passing, and a manual audit with representative qualitative examples (Sections 6–7).

2. Related Work

2.1 OCR Confidence and Failure Prediction

Engine confidence is the default verifier, and learned failure predictors regress the true-class probability from a classifier's own features [7]. Both score a single hypothesis in isolation and need access to engine internals; neither compares the claim with its visual competitors.

2.2 Post-OCR Correction with Large Models

Since 2024 post-OCR processing has moved to Large (Multimodal) language models. LLM post-correction gives "no free lunches": gains depend on language, model size and input error rate, and correct text can be damaged [8]. Vision–language models are also used as judges of recognition output, and fine-grained OCR benchmarks show that even the strongest

multimodal models still make frequent character-level errors [1]. Such systems target page-level reading, need large curated training sets and are orders of magnitude more expensive per word; we include a 7B judge [6] as a baseline.

2.3 End-to-end OCR Models and their Hallucinations

OCR-free document readers such as Donut [9] and VLM-based PDF linearisers such as olmOCR [10] read whole pages. Their errors are often *plausible*: multimodal LLMs hallucinate text on degraded images [2]. This motivates an engine-agnostic verifier that asks whether the image supports *this* string better than its near-miss rivals. Strong recognisers such as TrOCR [4] and PARSeq [3] are natural, frozen verifiers in this setting.

2.4 Contrast Against a Reference

Visual contrastive decoding reduces hallucination by contrasting a model's outputs on the real and on a distorted image [11]. Our render-normalised ratio applies the same principle with a reference *specific to each hypothesis*: a clean rendering of the very string being tested.

2.5 Graph Networks for OCR Verification

Character-level graphs comparing real and synthetic glyphs have been used to verify Hindi OCR [12], and Saxena *et al.* [13] represent each word as a relational line graph of real-versus-rendered character comparisons, verified few-shot by a GNN on a single private low-resolution dataset. Both require character segmentation of the word image and compare characters, not alternative readings. LOCUS needs no segmentation: its nodes are *hypotheses* and its edges are edit operations between them, encoded with relation-aware attention [14].

2.6 Pseudo-labels, Calibration and Label Shift

Training on pseudo-labels risks *confirmation bias* [15], commonly countered by confidence thresholding [16]; we test both (Section 6.7). Calibration degrades under distribution shift [17]; a new OCR engine acts like a label shift, for which EM prior re-estimation is a strong label-free correction [18], and conformal prediction offers distribution-free accept/review guarantees [19].

3. Proposed Method: LOCUS

Problem Statement: An OCR engine $\mathcal{E}$ maps a word (or text-line) image x to a claim $h = \mathcal{E}(x)$. The engine is a black box: we observe only h. With t the true transcription, verification predicts $y = \mathbf{1}[h = t]$ (CORRECT vs. INCORRECT) and, optionally, proposes a repair $\hat{t}$. We study the label-efficient regime with only $k \in \{5, 10, 25, 50, 100\}$ labelled words per class plus an unlabelled pool of images with engine outputs, and also report full supervision. Figure 2 gives an overview on a real test word and Algorithm 3.4 summarises inference.

3.1 Counterfactual String Graphs

For each claim h we build a graph $G(h) = (\mathcal{V}, \mathcal{E})$ whose nodes are strings: the claim and up to $M = 28$ counterfactual neighbours generated without any ground truth, in this priority order: (i) the greedy readings g_1, g_2 of two frozen verifier recognisers V_1, V_2; (ii) *reading-aligned edits*, the single substitutions, insertions and deletions that move h one step towards g_1 or g_2 under a Levenshtein alignment; (iii) *confusion edits*: substitutions within visually confusable groups (e.g. {O,0,o,Q,D}, {I,l,1,|}, {5,S}), case flips and single deletions. Every neighbour is linked to the claim by an edge typed by its edit operation (seven

types); *sibling* edges link neighbours that edit the same position (competing explanations of one glyph) and *path* edges link each reading to the single edits leading to it, giving a one-hot edge type $e_{uv} \in \{0,1\}^{11}$. OCR errors are near misses, so the truth lies inside $G(h)$ for most real errors (Supplementary Table S33), which makes repair possible.

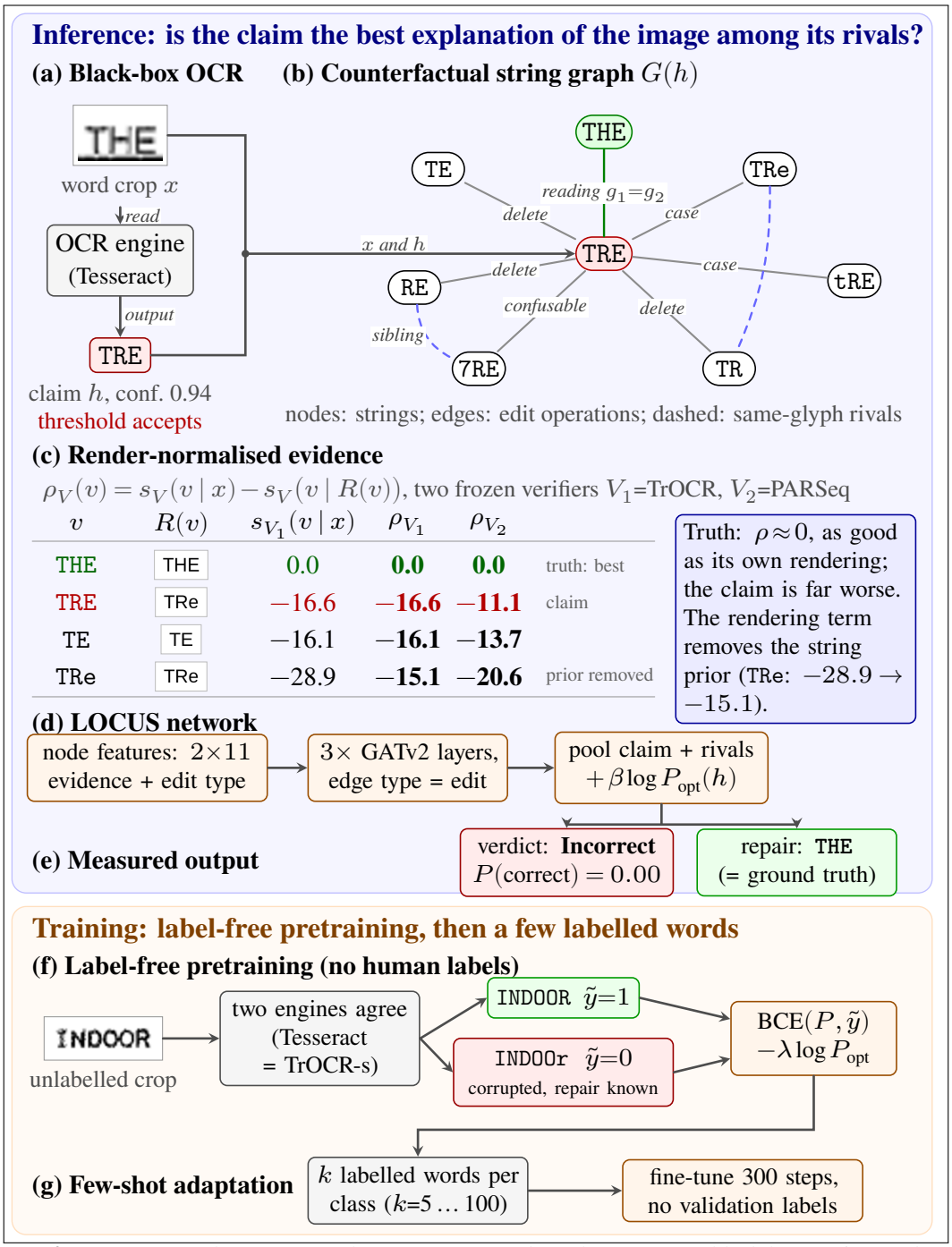

Figure 2. LOCUS at a glance, on a real FUNSD test word (truth THE). (a) A black-box engine reads TRE. (b) LOCUS builds the claim's counterfactual string graph. (c) Every hypothesis is scored by its render-normalised likelihood ratio under two frozen recognisers. (d) A relation-aware graph network compares the claim with its rivals and (e) rejects it, proposing the ground truth as repair. (f) Pretraining uses no human labels; (g) adaptation uses k labelled words per class. All strings, images and numbers shown are real.

3.2 Render-normalised Evidence

Let $s_V(v \mid x) = \sum_i \log p_V(v_i \mid v_{<i}, x)$ be the teacher-forced log-likelihood of string v under a frozen autoregressive recogniser V. It conflates visual fit with the prior that V's decoder has learned over strings: a correct but rare string (a serial number, a surname, lower-case text under a recogniser trained on upper-case receipts) scores low. With $R(v)$ a clean rendering of v in a standard font at the height of x, we define

$$\rho_V(v) \;=\; s_V(v \mid x) \;-\; s_V(v \mid R(v)). \tag{1}$$

Under the factorisation $\log p_V(v \mid x) \approx \log p_{\text{vis}}(x \mid v) + \log \pi_V(v) - \log Z(x)$ the string prior $\pi_V(v)$ appears in both terms and cancels; $\rho_V(v) \approx 0$ means "x is as convincing a picture of v as a perfect rendering", while strongly negative values flag visual disagreement (Figure 3). We also record the token-level minima of the log-likelihood and of the ratio (an error is often a single character) and the cosine distance between V's decoder states on x and on $R(v)$. Every quantity is computed with two recognisers of different design: V_1=TrOCR-base [4] (ViT encoder, BPE decoder; the printed, handwritten or scene-text checkpoint matching the domain) and V_2=PARSeq [3] (character-level permuted autoregressive decoder). Neither is fine-tuned. Each node thus carries 2×11 evidence features, seven claim-relative differences $\phi(v) - \phi(h)$, its edit type and its relative edit position.

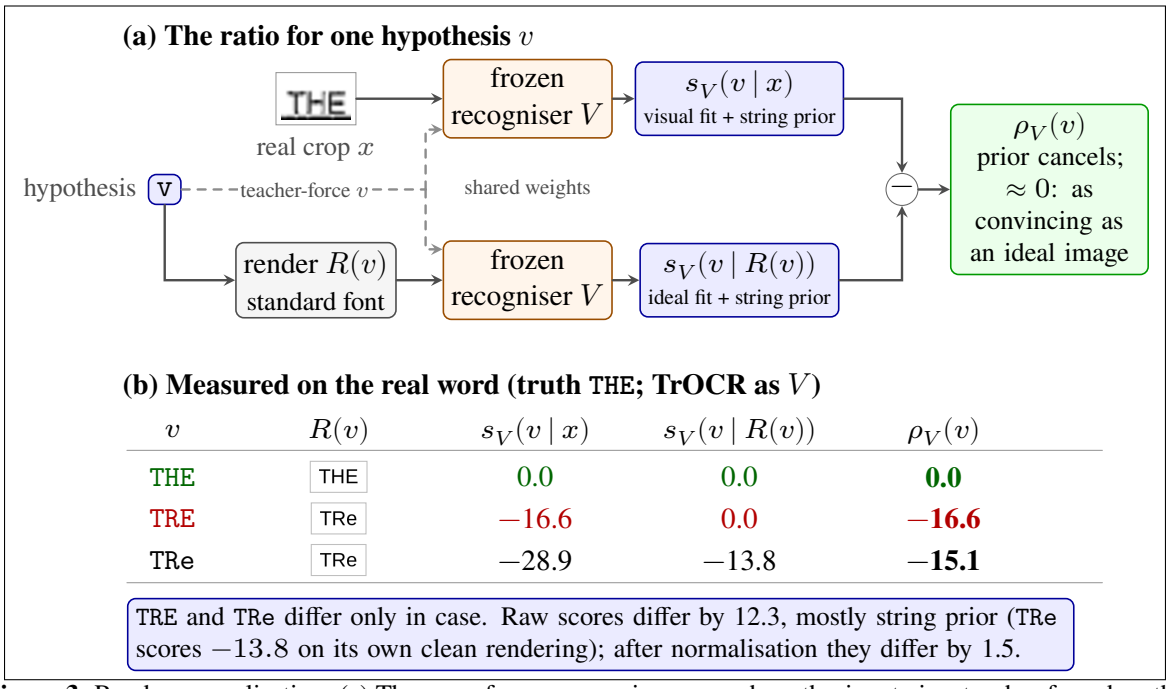

Figure 3. Render normalisation. (a) The same frozen recogniser scores hypothesis v twice, teacher-forced on the real crop and on a clean rendering of v; the difference cancels the decoder's prior over strings. (b) Real TrOCR values for the word of Figure 2: the case variant TRe looks far less likely than TRE on the raw score only because TrOCR's decoder dislikes the string, which the ratio removes.

Renderer and Preprocessing (Exact Configuration): $R(v)$ is drawn with the Pillow library in Liberation Sans Regular, a freely licensed font metrically equivalent to Arial, chosen *a priori* as a neutral sans-serif and never tuned on any test set. The canvas height equals the crop height clamped to $[32, 96]$ pixels; the font size is $0.75\times$ that height; the text is drawn black

on a white RGB canvas, vertically centred from its glyph bounding box, with a left margin of a quarter of the height and a total horizontal margin of half the height, and no blur, noise or augmentation. Real crop and rendering then pass through the *same* preprocessing of each recogniser: for TrOCR, bilinear (antialiased) resizing to 384×384 and normalisation $(x/255 - 0.5)/0.5$ in half precision; for PARSeq, bicubic resizing to 32×128 and the same normalisation. PARSeq scores v restricted to its 94-character set and truncated to 25 characters; TrOCR scores at most 60 BPE tokens. Both are scored teacher-forced from left to right, so $s_V(v \mid \cdot)$ is a sum of per-token log-probabilities. Because the ratio subtracts two scores of the *same* string by the *same* recogniser, a font or scale mismatch shifts both terms of a hypothesis similarly; Section 6.4 measures the residual sensitivity with six alternative renderings.

3.3 The LOCUS Network

Node features are standardised with statistics of the *unlabelled* pool and embedded linearly. Three relation-aware GATv2 layers [14] with edge features and residual LayerNorm update the node states:

$$\begin{aligned} h_u^{(\ell+1)} &= \mathrm{LN}\Big(h_u^{(\ell)} + \sigma\Big(\textstyle\sum_{v \in \mathcal{N}(u)} \alpha_{uv}^{(\ell)} W^{(\ell)} h_v^{(\ell)}\Big)\Big), \\ \alpha_{uv}^{(\ell)} &= \mathrm{softmax}_v\Big(a^\top \mathrm{LReLU}\big(W[h_u \| h_v \| e_{uv}]\big)\Big). \end{aligned} \tag{2}$$

Local-optimality Readout: A plausibility score $q_v = w_q^\top \mathrm{MLP}(h_v)$ defines $P_{\text{opt}}(v \mid G) = \exp q_v / \sum_{u \in \mathcal{V}} \exp q_u$, the probability that v is the best explanation of x within its neighbourhood. The verdict combines the claim's state, an attention pool and a max pool over its rivals, and the log-probability that the claim itself is the local optimum,

$$\mathrm{logit}\, P(y = 1 \mid x, h) = \mathrm{MLP}\big([h_h \,\|\, \textstyle\sum_v a_v h_v \,\|\, \max_v h_v]\big) + \beta \log P_{\text{opt}}(h \mid G), \tag{3}$$

and the same distribution gives the repair $\hat{t} = \arg\max_{v \neq h} P_{\text{opt}}(v \mid G)$. We also compute *character evidence* (per-position counterfactual margins of the claim under V_1, pooled by mean/min/max); it is used only by the "LR on character evidence" baseline and by ablations, not by LOCUS.

3.4 Label-free Pretraining and Few-shot Adaptation

From the unlabelled training images, we retain those for which two OCR engines produce identical transcriptions and pair each retained image with (i) the agreed string as a pseudo-correct claim and (ii) an OCR-style corruption of it (confusable substitution, case flip, deletion or insertion) as a pseudo-incorrect claim whose repair is the original. LOCUS is trained on these graphs with $\mathcal{L} = \mathrm{BCE}\big(P(y = 1 \mid x, h), \tilde{y}\big) - \lambda \log P_{\text{opt}}(\tilde{t} \mid G)$ ($\lambda = 0.5$), where $\tilde{y}$ is the pseudo-label and $\tilde{t}$ the node holding the pseudo-truth; no human label is used. Given k labelled words per class, the pretrained network is fine-tuned for 300 steps (AdamW, lr $3 \cdot 10^{-4}$, batch of half the support set, class-balanced binary cross-entropy plus the same repair term on correct support words). This is repeated for five independently drawn support sets and the five fine-tuned networks are averaged (Section 4); the average is the reported LOCUS verdict, thresholded at 0.5. Two alternative heads, a logistic probe on the frozen embedding and a prior-anchored logistic head, are reported only as ablations (Table 4). Feature standardisation uses statistics of the unlabelled training pool, and no validation labels are used in the few-shot regime.

Algorithm 1: LOCUS inference for one claim.

Input: word image x, engine claim h; frozen recognisers V_1, V_2; renderer R; five fine-tuned LOCUS networks.

Output: $P(\textsc{Correct} \mid x, h)$ and a repair proposal $\hat{t}$.

1. $g_1 \leftarrow V_1(x)$, $g_2 \leftarrow V_2(x)$ *(greedy readings)*
2. $\mathcal{V} \leftarrow \{h, g_1, g_2\} \cup \text{AlignedEdits}(h, g_1, g_2) \cup \text{ConfusionEdits}(h)$, truncated to $M{+}1$ nodes
3. for each $v \in \mathcal{V}$ and $V \in \{V_1, V_2\}$: $\rho_V(v) \leftarrow s_V(v \mid x) - s_V(v \mid R(v))$, plus token minima and latent distance
4. build typed edges (edit, sibling, path) and node features $[\phi(v) \,\|\, \phi(v) - \phi(h) \,\|$ edit type, position$]$
5. run 3 GATv2 layers; compute $P_{\text{opt}}(\cdot \mid G)$ and logit $P(y = 1 \mid x, h)$
6. average $P(y = 1 \mid x, h)$ over the networks fine-tuned on the five support sets
7. return P, and $\hat{t} = \arg\max_{v \neq h} P_{\text{opt}}(v \mid G)$ if $P < 0.5$

Why Verification does not Need the Truth Inside the Graph: Local optimality is a *necessary* condition for correctness: if $h = t$, no edit of h explains the image better, so (up to recogniser noise) no rival has $\bar{\rho}(v) > \bar{\rho}(h)$, with $\bar{\rho} = \rho_{V_1} + \rho_{V_2}$. Rejection therefore needs only *one* rival that beats the claim, not the truth. Because s_V is a sum over positions, an edit that moves an erroneous character h_i towards what is written raises $\bar{\rho}$ mainly through position i; a claim with several wrong characters thus loses to its partially corrected neighbours and to differing recogniser readings, even when the truth is several edits away and absent from $G(h)$, and a strongly negative $\rho(h)$ flags a poor fit with no rival at all. Repair, in contrast, *does* require $t \in G(h)$; Section 6.6 confirms this asymmetry.

Computational Cost: For $n = |\mathcal{V}|$ hypotheses (15–21 on average), evidence extraction runs, per recogniser, one encoder pass on the crop, n on the renderings and $2n$ teacher-forced decoder passes: $O(n\,C_{\text{rec}})$. Graph construction manipulates strings only, and the GNN costs $O(L(|\mathcal{E}|d + nd^2)) \approx 10^6$ operations per word ($L = 3$, $d = 64$), five orders of magnitude below one ViT-base encoder pass. The cost of LOCUS is the cost of its evidence, linear in n; Section 5.6 measures it and trades it off against accuracy.

4. Experimental Setup

4.1 A Verification Benchmark with Real OCR Errors

Public OCR datasets provide images and transcriptions but no verification labels. We create them by running real OCR engines and labelling each output against the ground truth (Supplementary Figure S9a), so every INCORRECT example is a genuine engine error, not a synthetic perturbation. Five public datasets cover four domains (Table 1, Supplementary Figure S8): FUNSD [20], word boxes in noisy scanned forms (the Hugging Face release stores boxes on the LayoutLM 0–1000 grid; they must be rescaled to pixels before cropping); CORD [21], words in camera-captured receipts (official train and validation splits form the training pool); SROIE [22], text lines of scanned receipts, whose ground truth is entirely upper-case, so comparison is case-insensitive; IAM, handwritten English words; and IIIT5K, scene-text words, for which we follow the standard case-insensitive alphanumeric protocol and give *every* verifier, including LOCUS and the VLM judge, the claim in this canonical form. Official splits are kept and pools are capped at 8,000 training and 4,000 test crops (fixed seed). The main engine is Tesseract 5 [23] (LSTM, `psm 7`); on handwriting Tesseract is degenerate (93% word

error), so IAM verifies TrOCR-small-handwritten. A second engine, TrOCR-small, is used for engine transfer. Test error rates span 29–47%.

Data Partitioning and Leakage Control: We keep the official splits, which are document-disjoint for FUNSD, CORD and SROIE, so all crops of a document fall in one split. Pretraining, support sets, thresholds and all learned baselines use only the *training* pool; no test crop, label or statistic enters pretraining, model selection or any threshold. A mechanical audit confirms that 0 pretraining images (2,561 FUNSD, 3,000 CORD, 2,663 SROIE, 3,000 IAM, 860 IIIT5K) come from a test split, and pixel hashing finds no train/test duplicates except 79 CORD crops from receipts duplicated across the official splits; removing them leaves LOCUS' macro-F1 (.978) and AUROC (.993) unchanged. The public IAM word release is split at word level, so it is not writer-disjoint. Hyperparameters were fixed on FUNSD and CORD, whose test sets were consulted during development, and applied unchanged to the held-out SROIE, IAM and IIIT5K.

Table 1. Benchmark: Five public datasets with real OCR errors. Error rate: share of engine outputs that differ from the ground truth (exact match; SROIE case-insensitive; IIIT5K case-insensitive alphanumeric).

Dataset	domain	unit	#train	#test	avg. len.	engine	err. train (%)	err. test (%)
FUNSD	scanned forms	word	8,000	4,000	5.1	Tesseract 5	28.7	36.9
CORD	camera receipts	word	8,000	2,356	4.9	Tesseract 5	37.0	39.9
SROIE	scanned receipts	line	8,000	4,000	11.5	Tesseract 5	47.8	47.0
IAM	handwriting	word	8,000	4,000	4.2	TrOCR-small	48.4	46.4
IIIT5K	scene text	word	2,000	2,999	5.0	Tesseract 5	14.2	29.0

What Counts as an Error: Labels inherit the datasets' conventions and noise. A share of incorrect claims differ from the ground truth only in case or punctuation (mostly fragments of neighbouring glyphs inside the word box), and our manual audit finds annotation errors and clipped crops among them (Section 7). We keep these claims (an engine that emits `Barbara |` for the field `Barbara` is wrong for a downstream consumer) but also report every method on genuine character errors only (Section 6.5). Supplementary Table S33 shows that the counterfactual graph contains the truth for 83–97% of real errors on printed and scene text and for 55% on handwriting, where many errors span several characters.

4.2 Baselines, Protocol and Metrics

All baseline methods are evaluated on the same test words, and every threshold is fitted on the same k-shot support set as LOCUS (maximising macro-F1).

Engine Confidence: Tesseract's mean word confidence or TrOCR-small's mean token probability (temperature scaling is monotone and leaves ranking metrics unchanged).

ConfidNet-style Learned Confidence: [7] an MLP trained on the support set from the engine confidence and the verifier's likelihood statistics of the claim.

Re-recognition Agreement: accept iff an independent recogniser reads the same string: TrOCR-base [4], PARSeq [3] and Florence-2 [5]; for PARSeq we also report a *graded* variant ranked by its sequence confidence (agree $\rightarrow +c$, disagree $\rightarrow -c$), the strongest re-recognition baseline.

Two-engine Voting: accept iff Tesseract and TrOCR-small agree.

VLM Judge: Qwen2.5-VL-7B-Instruct [6] asked whether the image shows exactly the claimed text, scored by $P(\texttt{yes})/(P(\texttt{yes})+P(\texttt{no}))$ at the natural 0.5 threshold and at an *oracle* threshold tuned on the test set.

Strong learned Baselines on Our Evidence: logistic regression and gradient-boosted trees on flattened LOCUS evidence and/or character evidence, DeepSets over rivals (no edges), and LOCUS from scratch.

For each k we draw five class-balanced support sets (seeds 0–4); the test set keeps its natural class ratio, and we report the ensemble of the five models. Metrics: macro-F1 (primary), AUROC, AURC (area under the risk–coverage curve, lower is better), coverage@95 (share of words that can be accepted while the accepted set stays $\geq 95\%$ correct) and top-1 repair accuracy. In every column of a comparison table the best entry is bold on green and the second best underlined on blue.

Implementation: hidden size 64, 3 GATv2 layers with 4 heads, dropout 0.2; pretraining 15 epochs (AdamW, lr 10^{-3}, batch 64); rendering in Liberation Sans; the trainable GNN has 0.05 M parameters and evidence extraction runs at ≈18 words/s on one RTX A6000 (Section 5.6).

Reproducibility: All numbers are produced by scripts from raw per-run result files. Code, benchmark manifests (crop coordinates, splits and claims; images and ground truth must be obtained from the dataset owners), per-run results and a step-by-step reproduction guide (Supplementary Section S8) accompany the article.

5. Results and Discussion

5.1 Main comparison

Tables 2 and 3 compare LOCUS with all baselines at $k = 25$ (other k and all metrics: Supplementary Section S6).

Table 2. Main comparison, macro-F1 at $k = 25$ labelled words per class (learned methods: ensemble of five seeds; threshold baselines: mean over the five supports). FUNSD and CORD were used for development; SROIE, IAM and IIIT5K are held out. †Threshold tuned on the test set (upper bound); ‡the best baseline chosen separately for every dataset in hindsight (both excluded from the colour marks).

Method	FUNSD	CORD	SROIE	IAM	IIIT5K	Avg.
Baselines						
Engine MSP confidence	.756	.777	.780	.802	.788	.781
ConfidNet-style learned conf. [7]	.771	.842	.826	.801	.922	.832
Two-engine voting	.609	.772	.976	.759	.862	.796
Re-OCR: TrOCR-base [4]	.605	.811	.979	.758	.952	.821
Re-OCR: Florence-2 [5]	.647	.615	.478	.369	.823	.586
Re-OCR: PARSeq [3]	.946	.973	.742	.879	.993	.907
Re-OCR: PARSeq, graded	.946	.973	.742	.879	.993	.907
VLM judge: Qwen2.5-VL-7B [6]	.760	.891	.866	.700	.888	.821
VLM judge, oracle threshold†	.762	.902	.883	.719	.928	.839
Best baseline per dataset, chosen in hindsight ‡	.946	.973	.979	.879	.993	.954

Strong learned baselines on our evidence						
LR on flattened graph evidence	.948	.955	.955	.904	.969	.946
LR on character evidence	.935	.971	.959	.887	.965	.943
hline GBDT on all evidence	.922	.961	.962	.893	.955	.939
DeepSets over rivals (no edges)	.942	.977	.968	.909	.965	.952
LOCUS-GNN from scratch	.933	.969	.961	.908	.962	.947
Ours						
LOCUS zero-shot (no labels)	.918	.904	.798	.825	.968	.883
LOCUS (k-shot)	.943	.978	.973	.906	.971	.954

Table 3. Main comparison, AUROC at $k = 25$ (same protocol as Table 2).

Method	**FUNSD**	**CORD**	**SROIE**	**IAM**	**IIIT5K**	**Avg.**
Baselines						
Engine MSP confidence	.861	.892	.859	.878	.938	.886
ConfidNet-style learned conf. [7]	.845	.929	.912	.861	.980	.905
Two-engine voting	.680	.807	.976	.767	.905	.827
Re-OCR: TrOCR-base [4]	.675	.841	.978	.766	.966	.845
Re-OCR: Florence-2 [5]	.715	.685	.578	.522	.879	.676
Re-OCR: PARSeq [3]	.939	.975	.763	.885	.991	.911
Re-OCR: PARSeq, graded	.976	.986	.760	.914	.995	.926
VLM judge: Qwen2.5-VL-7B [6]	.846	.945	.941	.788	.974	.899
VLM judge, oracle threshold†	.846	.945	.941	.788	.974	.899
Best baseline per dataset, chosen in hindsight ‡	.976	.986	.978	.914	.995	.970
Strong learned baselines on our evidence						
LR on flattened graph evidence	.975	.995	.990	.970	.991	.984
LR on character evidence	.973	.992	.990	.959	.993	.981
GBDT on all evidence	.969	.988	.991	.964	.990	.981
DeepSets over rivals (no edges)	.971	.994	.992	.972	.990	.984
LOCUS-GNN from scratch	.969	.992	.990	.969	.990	.982
Ours						
LOCUS zero-shot (no labels)	.949	.977	.966	.909	.983	.957
LOCUS (k-shot)	.974	.993	.992	.969	.991	.984

General-purpose Verifiers are far from Sufficient: Engine confidence, its learned refinement, engine voting, Florence-2 re-recognition and the 7B VLM judge all trail LOCUS by large margins on every dataset. Despite 8.3B parameters the VLM judge reaches only macro-F1 on FUNSD and on IAM, and an oracle threshold barely helps: its errors are errors of *ranking*, not of calibration. Character-exact judgements (`O` vs. `0`, a missing comma, a case flip) are what instruction-tuned VLMs gloss over.

Re-recognition with a Strong Recogniser is the Baseline to Beat: Accepting a claim iff PARSeq reads the same string is by far the strongest baseline, and its graded variant more so. LOCUS improves on it where PARSeq is fallible: on IAM handwriting by macro-F1 and AUROC, and on CORD receipts (macro-F1 vs. ; AUROC gain significant, Table 9). On FUNSD the two are statistically tied, and on IIIT5K, the scene-text benchmark on which PARSeq was designed and reads almost every word correctly, PARSeq agreement remains better (vs.). On

held-out SROIE the ranking of baselines flips: TrOCR's printed checkpoints were fine-tuned on SROIE receipts, so TrOCR re-recognition is strong (.979 macro-F1) while PARSeq agreement fails on these long receipt lines (.742); LOCUS is slightly below the former in macro-F1 (.973) and above it in AUROC (.992 vs. .978, Table 9). No single re-recognition baseline is the best choice on every dataset (PARSeq on forms and scene text, TrOCR on SROIE, neither on handwriting), whereas LOCUS is within 0.006 macro-F1 of the best baseline or better on four of the five datasets. Its average macro-F1 (.954) is the highest of all methods in Table 2: well above the best single baseline (.907, PARSeq), equal to choosing the best baseline separately for every dataset in hindsight (.954), and marginally above simple learners trained on our own counterfactual evidence (.939–.952).

Learned Baselines on Our Evidence and the Label-free Model: Logistic regression, gradient boosting and DeepSets on the *same* counterfactual evidence are competitive with LOCUS (middle blocks): most of the gain over general-purpose baselines comes from the counterfactual, render-normalised *evidence* itself, and Section 6.3 shows that message passing adds little on top of it. Pretrained only on engine agreement and synthetic corruptions, zero-shot LOCUS already exceeds engine confidence, learned confidence, Florence-2 and the VLM judge in AUROC on every dataset, and engine voting on four of five (all except SROIE, where both voting engines include a SROIE-trained recogniser).

5.2 Label Efficiency and Render Normalisation

Supplementary Figure 10 shows that LOCUS is close to its fully supervised accuracy with $k = 10$–25 labelled words per class on the document datasets, and keeps improving with more labels on IAM; seed variance shrinks rapidly with k (Supplementary Section 6). Supplementary Figure S7 and Table S17 isolate Equation (1): with TrOCR as the only verifier it helps where TrOCR's prior is mismatched (SROIE .956 vs. .902) and is neutral with PARSeq alone; render normalisation corrects a *biased* recogniser and is neutral for an unbiased one.

Why does Removing Render Normalisation Cost Only 0.002?: The average hides where the ratio acts (Supplementary Table S34). Where TrOCR's string prior is mismatched it is large: TrOCR AUROC on SROIE claims with digits rises from .901 to .976, and on lower- and mixed-case words from .714 to .807 (FUNSD), .752 to .948 (CORD) and .722 to .972 (SROIE); on upper-case words it changes nothing, and on handwriting it hurts (IAM digits .740 to .638), since a printed rendering is a poor reference there. In the full model PARSeq and the agreement features already cover the subsets where the ratio helps, hence the small ablation effect. The ratio is what makes a single, domain-mismatched recogniser usable as a verifier.

5.3 Ablations

Table 4. Ablations at $k = 25$, macro-F1 (ensemble of five seeds). Δ: mean change w.r.t. the full model over the datasets where the variant was run (AUROC: Supplementary Table S8). The "LOCUS (full)" row is an independent re-run of the model of Table 2; the last two rows give the range over three further independent re-runs (run-to-run noise).

Variant	FUNSD	CORD	SROIE	IAM	IIIT5K	Δ
LOCUS (full)	.939	.979	.974	.902	.970	
+ CV-selected prior/anchored head	.944	.961	.965	.903	.972	-0.004
+ linear probe on embedding	.943	.970	.965	.901	.967	-0.004

w/o verifier-agreement node features	.937	.978	.963	.902	.958	-0.005
w/o edit-graph GNN (char evidence LR)	.935	.971	.959	.887	.965	-0.009
w/o label-free pretraining	.932	.969	.962	.908	.962	-0.006
w/o local-optimality term	.943	.980	.968	.899	.967	-0.002
w/o repair loss	.941	.981	.974	.899	.965	-0.001
w/o render-normalisation	.942	.977	.967	.893	.973	-0.002
w/o latent render distance	.940	.978	.971	.914	.972	+0.002
likelihood only (no render)	.940	.979	.969	.913	.975	+0.002
TrOCR verifier only	.942	.980	.971	.881	.975	-0.003
PARSeq verifier only	.938	.978	.969	.898	.970	-0.002
star edges only	.943	.981	.968	.898	.972	-0.001
no message passing	.945	.978	.969	.895	.979	+0.000
GCN backbone	.942	.978	.965	.905	.969	-0.001
GraphSAGE backbone	.933	.978	.970	.906	.966	-0.002
GIN backbone	.940	.972	.955	.889	.964	-0.009
Run-to-run noise: range over three independent re-runs						
LOCUS (full), 3 independent re-runs	.940–.941	.978–.980	.973–.973	.896–.903	.971–.972	
no message passing, 3 re-runs	.944–.945	.978–.978	.969–.970	.894–.895	.979–.979	

Table 4 reports every component. The clearest effects are label-free pretraining and operating on the edit graph at all (versus the same character evidence without rival hypotheses). Several conceptually central components are individually *redundant* in the full model: removing message passing, the local-optimality term, the repair loss or render normalisation changes macro-F1 by less than a point on most datasets, because the two verifiers and the agreement features carry overlapping information; GCN and GraphSAGE backbones perform like GATv2 and GIN is slightly worse (-0.009). The value of LOCUS lies in *what* is compared (the claim against its counterfactual rivals, under prior-cancelled evidence) more than in the network that compares them.

Consistency of the Reported Numbers: The full model appears in Tables 2 and 4 and Supplementary Table S43 with slightly different values (FUNSD .943, .939, .943) because each comes from an independent run of the whole pipeline; seeds are fixed, but GPU scatter kernels are not bit-deterministic. Three further re-runs (Table 4, bottom) span at most 0.006 macro-F1 and 0.001 AUROC. Table 2 is the reference result, and effects below this noise level are not interpreted; repeated "no message passing" runs differ from the full model by -0.005 to $+0.007$ macro-F1, with changing sign.

5.4 Transfer Across Engines and Datasets

A verifier that must be retrained for every engine is of limited use. Trained on Tesseract claims and tested on TrOCR-small claims (and vice versa), the fully supervised *ranking* transfers with an AUROC loss of at most 0.006 on FUNSD, CORD and IIIT5K (at most 0.023 at $k = 25$), but SROIE loses 0.02–0.07, and the 0.5 decision threshold transfers poorly everywhere because the two engines' error rates and types differ; a new engine therefore needs at least threshold re-calibration (Supplementary Table S35). The ranking transfers

well across all twenty dataset pairs (AUROC .892–.995; e.g. FUNSD→CORD .995 vs. .996 in-domain, CORD→FUNSD .968 vs. .979). The weakest transfers are those *to* handwriting (SROIE→IAM .892, CORD→IAM .954) and from receipt lines to forms (SROIE→FUNSD .941). As for engines, macro-F1 at the fixed 0.5 threshold transfers much less well than the ranking (e.g. SROIE→FUNSD .620).

Re-calibrating the Threshold for a New Engine: (Table 5). A new engine changes the error rate (FUNSD: 37% to 62%) and the error types. Label-free EM prior re-estimation with bias-corrected calibration [18] does *not* help: it lowers macro-F1 in six of eight directions and usually increases calibration error, because the class-conditional scores also shift. Re-fitting the threshold on 25 labelled target words per class recovers most of the loss where the transferred threshold fails (FUNSD .689 to .862, CORD .728 to .931 and .836 to .935, SROIE .767 to .853) but is noisier where it already works; 5 words per class are too few. Re-adapting LOCUS on 25 target words per class is best in five of eight directions and close to the best in two others (the exception, SROIE with TrOCR-s claims, has only 6.3% errors). We therefore recommend treating a new engine as a new domain: label about 25 words per class, re-adapt, and monitor accepted precision; conformal calibration on that sample would add a distribution-free guarantee [19].

Table 5. Threshold re-calibration under engine transfer (LOCUS, macro-F1 on the target engine). err.: target error rate (%). EM: label-free prior-shift EM. 5/25: target threshold fitted on 5 or 25 labelled target words per class. In-engine: LOCUS adapted on the target engine ($k = 25$). ECE: expected calibration error at 0.5 / after EM.

Dataset	Train→Test	err.	0.5	EM	5	25	in-engine	ECE
FUNSD	Tess.→TrOCR-s	61.7	.689	.573	.817	.862	.955	.235/.359
FUNSD	TrOCR-s→Tess.	36.9	.946	.943	.907	.926	.940	.031/.028
CORD	Tess.→TrOCR-s	37.4	.728	.575	.912	.931	.957	.205/.276
CORD	TrOCR-s→Tess.	39.9	.836	.836	.870	.935	.978	.137/.137
SROIE	Tess.→TrOCR-s	6.3	.752	.625	.497	.559	.639	.036/.052
SROIE	TrOCR-s→Tess.	47.0	.767	.735	.841	.853	.973	.239/.271
IIIT5K	Tess.→TrOCR-s	28.3	.984	.983	.866	.953	.990	.015/.007
IIIT5K	TrOCR-s→Tess.	29.0	.968	.971	.886	.949	.970	.026/.020

5.5 Practical Use: Human Review, Repair and Cost

The operational question in a digitisation pipeline (Supplementary Figure S9b) is how many words a human must check to guarantee the quality of the rest (Supplementary Figure S11, Supplementary Table S45). At 95% accepted precision, LOCUS requires reviewing 44.7% of SROIE lines (TrOCR re-recognition 45.0%, PARSeq 69.5%, the VLM judge 76.2%, engine confidence 90.9%) and 51.4% of IAM words (PARSeq 54.6%, the VLM judge 94.8%). On FUNSD, CORD and IIIT5K, LOCUS and PARSeq are within two points of each other (FUNSD 37.2% vs. 35.4%, CORD 37.4% vs. 38.6%, IIIT5K 25.6% vs. 25.3%), while engine confidence and the VLM judge require 1.6–2.3× more review than LOCUS on the three receipt and form datasets. At the strictest target (98%) PARSeq is better on FUNSD and IAM. Verification and repair are separate tasks, and LOCUS addresses the first. For repair (Table 6), the truth lies in the counterfactual graph for most errors, but substituting the better verifier's reading is at least as good a top-1 repair as LOCUS' label-free proposal on every dataset: LOCUS is a verifier first, and its repair head, trained only on synthetic corruptions, is a useful by-product rather than a state-of-the-art corrector. We therefore do not claim repair as a contribution. The repair

loss does not affect verification either (removing it changes macro-F1 by -0.001, Table 4). In deployment the two roles combine naturally: LOCUS decides which words are sent for review, and the reviewer is shown the reading of the stronger verifier as the suggested correction. Repair is in any case bounded by graph coverage, which is analysed with multi-character errors in Section 6.6. Scoring $\sim$17 hypotheses with two recognisers and two renderings runs at about 18 words/s on one GPU, slower than a single PARSeq pass and about $3\times$ slower than a batched 7B judge, although its frozen models are $23\times$ smaller (Table 6); Section 5.6 analyses this cost and how to reduce it.

Table 6. Repair and cost. Top: share of real errors whose truth lies in $G(h)$, and top-1 exact repair accuracy (%) of substituting TrOCR's reading, PARSeq's reading, or LOCUS' label-free proposal. Bottom: measured throughput on 256 FUNSD words on one RTX A6000.

Dataset	errors	in $G(h)$	TrOCR	PARSeq	LOCUS
FUNSD	1474	82.8	31.3	72.4	67.7
CORD	941	94.9	70.4	89.5	81.4
SROIE	1880	92.6	89.5	44.1	89.5
IAM	1857	55.4	14.4	46.5	29.0
IIIT5K	869	97.0	86.8	95.3	93.6

Component	parameters	words/s
LOCUS evidence extraction (TrOCR-base + PARSeq, both renderings)	358M (frozen)	17.9
LOCUS GNN (trainable)	0.05M	–
PARSeq re-recognition	24M	372.3
Florence-2-base re-recognition	0.23B	14.9
Tesseract 5 (CPU, 1 thread)	–	9.3
Qwen2.5-VL-7B judge	8.3B	53.7

5.6 Computational Cost, Graph Size and a Two-stage Cascade

Table 7. Graph size and composition ($k = 25$, macro-F1 / AUROC). nodes: average hypotheses per claim (FUNSD); w/s: measured words/s; cov.: share of IAM errors whose truth is in the graph (%).

Graph	nodes	w/s	FUNSD	CORD	SROIE	IAM	IIIT5K	cov.
M=2	3.0	–	.937/.962	.980/.993	.969/.991	.901/.962	.972/.991	52.0
M=4	4.9	49	.943/.969	.977/.993	.966/.991	.901/.964	.972/.991	53.9
M=8	8.6	32	.940/.964	.973/.992	.965/.991	.904/.966	.972/.991	54.6
M=16	13.7	21	.943/.972	.980/.993	.969/.992	.894/.965	.972/.990	55.1
M=28 (default)	17.2	17	.940/.974	.979/.993	.972/.992	.899/.967	.970/.991	55.4
readings only	1.9	–	.942/.960	.978/.992	.966/.990	.899/.960	.974/.990	50.8
edits only (no readings)	13.4	–	.945/.970	.976/.993	.965/.992	.899/.965	.961/.987	4.1

Where the Time Goes: On 512 FUNSD words (one RTX A6000), graph construction runs at words/s and the GNN at words/s (edges per graph): both are negligible. The time is spent scoring $\sim$17 hypotheses with two recognisers on the crop and on each rendering ($\approx$70 recogniser passes per word); larger batches do not help (17.0 vs. 16.9 words/s for batch 32 and 128), and capping the graph at $M = 16$, 8 or 4 rivals gives 21, 32 and 49 words/s.

Graph Size and Neighbour Selection: (Table 7). Accuracy is almost flat from $M = 28$ to $M = 4$ (largest macro-F1 change 0.007), and $M = 2$ loses at most 0.012 AUROC. The

recogniser readings carry most of the value: without them the share of IAM errors with the truth in the graph falls from 55.4% to 4.1%. The cap binds mainly for long SROIE lines (44.9% of claims; 7–18% elsewhere), and accuracy decreases with length (SROIE .992 for 1–3 characters to .959 for 16 or more; Supplementary Table S4). $M = 8$ is a safe choice when throughput matters.

A Two-stage Cascade: Most words are easy, so we accept a claim directly when PARSeq (372 words/s) reads exactly the claim with confidence ≥ 0.95 and run LOCUS only on the rest (Table 8). Only 34–73% of words reach LOCUS, early-accepted words have 0.4–3.1% errors, macro-F1 changes by -0.001 to $+0.016$, and throughput rises to 23–46 words/s (1.3–2.6×); with $M = 8$ it would exceed 40 words/s except on SROIE. LOCUS is thus a second-stage verifier for the uncertain fraction of a stream, not a replacement for a fast reader. The cascade also closes most of the IIIT5K gap to PARSeq (.993), most of which comes from claims correct only under the case-insensitive protocol (Section 7): it reaches .987 macro-F1 and .995 AUROC, equal to the best AUROC on IIIT5K (PARSeq graded).

Table 8. Cascade ($k = 25$): accept when PARSeq reads the claim with confidence ≥ 0.95, otherwise run LOCUS. routed: share of words sent to LOCUS; early err.: error rate among early-accepted words (%); w/s: effective words/s.

Dataset	LOCUS on all words		cascade				
	F1	AUROC	routed	early err.	F1	AUROC	w/s
FUNSD	0.943	0.974	43.2	3.1	0.943	0.977	37
CORD	0.978	0.993	49.7	0.4	0.979	0.995	33
SROIE	0.973	0.992	73.0	1.9	0.972	0.988	23
IAM	0.906	0.969	68.5	1.7	0.908	0.973	24
IIIT5K	0.971	0.991	34.1	0.4	0.987	0.995	46

6. Analysis: What Drives the Gains?

6.1 Statistical Significance at the Document Level

Words cropped from one page share scanner, font and annotator, so a word-level bootstrap overstates confidence. For FUNSD and CORD we recovered the source document of every crop (by replaying the seeded crop selection, verified against the released manifest with zero mismatches) and resample whole *documents* (50 test forms, 100 test receipts). IIIT5K has one word per image, so word-level resampling is exact; the public SROIE and IAM releases carry no document identifiers, so their intervals are word-level and may be optimistic (Table 9). Against the strongest baseline, LOCUS is significantly better on IAM in both metrics (ΔAUROC $+0.055$, ΔF1 $+0.026$, $p < 0.001$), better in AUROC on CORD at the document level ($+0.007$, $p = 0.023$; macro-F1 $+0.005$, not significant) and on SROIE ($+0.014$, $p < 0.001$, while TrOCR re-recognition is slightly better in macro-F1, -0.006, $p = 0.002$); it is statistically tied with PARSeq on FUNSD at the document level ($p = 0.25$ and $p = 0.12$) and significantly worse than PARSeq on IIIT5K (-0.004 AUROC, -0.021 macro-F1). It is significantly better than the VLM judge and than TrOCR re-recognition on every dataset except SROIE, where the latter is in-domain.

Table 9. Paired bootstrap (2,000 resamples) of LOCUS ($k = 25$) against the strongest baselines: difference in AUROC and in macro-F1, 95% interval and two-sided p. Resampling unit: documents for FUNSD and CORD, words otherwise.

Dataset	unit	Baseline	ΔAUROC	95% CI	p	ΔF1	95% CI	p
FUNSD	50 docs	PARSeq re-OCR (graded)	-0.002	[-0.007, +0.002]	0.252	-0.004	[-0.008, +0.001]	0.121
FUNSD	50 docs	TrOCR-base re-OCR	+0.299	[+0.266, +0.330]	<0.001	+0.338	[+0.287, +0.389]	<0.001
FUNSD	50 docs	Qwen2.5-VL-7B judge	+0.128	[+0.111, +0.146]	<0.001	+0.183	[+0.168, +0.198]	<0.001
CORD	100 docs	PARSeq re-OCR (graded)	+0.007	[+0.001, +0.015]	0.023	+0.005	[-0.001, +0.012]	0.122
CORD	100 docs	TrOCR-base re-OCR	+0.152	[+0.122, +0.184]	<0.001	+0.167	[+0.128, +0.206]	<0.001
CORD	100 docs	Qwen2.5-VL-7B judge	+0.049	[+0.036, +0.063]	<0.001	+0.088	[+0.071, +0.105]	<0.001
SROIE	4000 words	PARSeq re-OCR (graded)	+0.232	[+0.217, +0.246]	<0.001	+0.232	[+0.218, +0.246]	<0.001
SROIE	4000 words	TrOCR-base re-OCR	+0.014	[+0.010, +0.017]	<0.001	-0.006	[-0.010, -0.002]	0.002
SROIE	4000 words	Qwen2.5-VL-7B judge	+0.051	[+0.044, +0.058]	<0.001	+0.107	[+0.097, +0.118]	<0.001
IAM	4000 words	PARSeq re-OCR (graded)	+0.055	[+0.047, +0.063]	<0.001	+0.026	[+0.015, +0.037]	<0.001
IAM	4000 words	TrOCR-base re-OCR	+0.203	[+0.192, +0.214]	<0.001	+0.148	[+0.134, +0.161]	<0.001
IAM	4000 words	Qwen2.5-VL-7B judge	+0.181	[+0.169, +0.194]	<0.001	+0.206	[+0.191, +0.222]	<0.001
IIIT5K	2999 words	PARSeq re-OCR (graded)	-0.004	[-0.007, -0.001]	0.005	-0.021	[-0.028, -0.016]	<0.001
IIIT5K	2999 words	TrOCR-base re-OCR	+0.025	[+0.019, +0.031]	<0.001	+0.019	[+0.010, +0.027]	<0.001
IIIT5K	2999 words	Qwen2.5-VL-7B judge	+0.017	[+0.012, +0.023]	<0.001	+0.083	[+0.072, +0.094]	<0.001

Correcting Word-level Intervals for Correlation: SROIE and IAM carry no document identifiers, so we bound within-document correlation on FUNSD and CORD, where both resampling units exist: the ratio of document- to word-level bootstrap variance implies an intra-document correlation of at most $\hat{\rho} = 0.254$. Applying this worst case to SROIE (347 receipts) and IAM (657 writers) inflates the variance by 3.67 and 2.29 (Supplementary Table S37). All conclusions survive except that the SROIE macro-F1 deficit against TrOCR (-0.005) is no longer significant (a tie); the AUROC gain remains.

Seed Level Comparison: When compared individually with the strongest baseline, all five models trained with $k = 25$ machieve improvements in AUROC on IAM ($+0.030$ to $+0.046$) and SROIE, and four of five on CORD; the IAM macro-F1 gain ($+0.027$), however, holds only for the ensemble (single models -0.016 to $+0.015$; Supplementary Table S5). We therefore state the IAM advantage as one in AUROC and, for macro-F1, of the ensemble.

6.2 How Much does LOCUS Depend on its Recognisers?

Supplementary Table S36 lists each verifier's word accuracy and LOCUS' gain over the better re-recognition baseline; Supplementary Table S42 splits every test set by *which* verifier reads the word correctly. Three regularities hold on all five datasets:

1. When both verifiers read the word correctly every method is nearly perfect.
2. When only one verifier is right, re-recognition built on the *other* verifier fails badly, while LOCUS stays accurate because it weighs the evidence of both (IAM: 89.5% vs. 25.7% for PARSeq agreement on words only TrOCR reads, 89.0% vs. 43.6% for TrOCR agreement on words only PARSeq reads).
3. When neither is right, all methods degrade. LOCUS' gain is therefore governed by how often the verifiers *disagree in correctness*: it is largest on IAM, where each verifier alone is right on only 42–65% of words, and vanishes on IIIT5K, where PARSeq is right on 98.5%.

LOCUS is best understood as a learned, label-efficient combination of complementary recognisers through their counterfactual evidence, not as a substitute for a strong in-domain recogniser.

6.3 Why Counterfactual Evidence Matters More than Message Passing

Removing message passing does not hurt (Table 4). To understand why, Supplementary Table S43 runs five controlled variants, each pretrained label-free with the same change. Three observations explain the null effect of message passing. First, removing message passing changes neither metric by more than 0.009 on any dataset (A vs. B). Second, the *explicit* comparison of each rival with the claim does matter: removing the claim-relative features costs 0.010 to 0.023 macro-F1 on CORD, SROIE and IIIT5K (A vs. C), and adding message passing back does not recover it (C vs. D). The network is not learning the comparison from the graph; it uses the comparison when it is computed for it. Third, the graph is a depth-one star whose readout already pools over every rival, so a message-passing layer can only re-express information the readout sees anyway. Removing the rivals entirely (E) costs ranking quality on FUNSD (AUROC .974 to .961) but little elsewhere, because the claim node keeps its own evidence and its agreement with both verifiers' readings, which summarise what the strongest rivals would say. In short, what helps is counterfactual *evidence*: scoring the claim against alternatives under prior-cancelled likelihoods and making the comparison explicit; the graph is a convenient container for it, not the source of the gain.

Separating the Evidence from the GNN: Table 2 decomposes the gain of LOCUS (.954 average macro-F1) over the best general-purpose baseline (PARSeq, .907). (i) Simple learners on the same counterfactual evidence and the same 25 words per class reach .939 (gradient boosting), .946 (logistic regression) and .952 (DeepSets): the evidence accounts for 70–95% of the gain. (ii) The graph network trained from scratch (.947) is no better. (iii) Label-free pretraining adds 0.007, the only consistent increment. In AUROC the simple learners equal LOCUS (.981–.984 vs. .984), and message passing contributes nothing beyond run-to-run noise. The GNN is therefore *not* necessary: a tabular or set-based classifier on the evidence is a valid, simpler implementation. We keep the graph because it holds a variable number of rivals with typed edit relations, supports pretraining with a repair objective and costs nothing at inference

(Section 5.6); the scientific contribution is the evidence.

6.4 Sensitivity to the Rendering

The ratio depends on a rendering $R(v)$ whose font never matches the document. We re-extracted all evidence with four other fonts (including a handwriting-like one) and two other glyph sizes on fixed subsets of FUNSD, CORD and IAM (1,000 training and 1,000 test words each) and re-evaluated the claim's ratio alone, the training-free ρ-margin rule, and LOCUS trained on the subset (Supplementary Table S38). The trained model is insensitive to the rendering: across seven renderings its macro-F1 varies by at most 0.005 on every dataset (FUNSD .937–.939, CORD .972–.977, IAM .919–.923), and the AUROC of the claim's ratio alone by at most 0.011. The training-free rule is more sensitive (AUROC .920–.940 on FUNSD, .919–.944 on IAM); it is best with a serif or monospaced font rather than our default sans-serif, which we did not tune, and a handwriting-like font does not help on handwriting. A mismatched rendering is thus absorbed by learning; only the label-free rule would benefit from choosing the font.

Latin-script Bias and Other Scripts: The renderer uses a Latin font and all benchmarks are Latin-script, so these results show robustness only to the choice of Latin font. The ratio itself is script-agnostic, but connected and complex scripts (Arabic, Devanagari, Bengali) need a shaping engine for rendering (e.g. HarfBuzz via Raqm) so that ligatures and conjuncts appear as in real text, a confusion model over grapheme clusters rather than code points, and script-specific recognisers, whose readings carry most of the value of the graph (Table 7). For historical documents a period-appropriate font is the natural reference. These settings are untested (Section 8).

6.5 Robustness to Annotation and Cropping Artefacts

Supplementary Table S39 re-scores every method after removing incorrect claims that differ from the ground truth only in case or punctuation, which the audit (Section 7) shows are often annotation conventions, label noise or fragments of neighbouring glyphs. The removed share is large on FUNSD (51% of errors) and moderate on CORD, SROIE and IAM (18–23%); IIIT5K's canonical protocol already excludes such differences. LOCUS' AUROC changes by at most 0.011 (it rises slightly or is unchanged), and its ordering relative to the baselines is unchanged except that the VLM judge, whose AUROC rises most (FUNSD .846 to .928, SROIE .941 to .982), is visibly penalised by these artefacts. The conclusions of Sec. 5.1 therefore do not rest on annotation conventions.

6.6 When the Truth is not in the Graph, and Correlated Recogniser Failures

Only 55.4% of IAM errors have their truth inside $G(h)$. This bounds repair, not verification: a claim is rejected when *any* rival explains the image better (Section 3). Table 10 confirms it. On IAM LOCUS rejects 90.6% of out-of-graph errors, more than the 83.7% of in-graph ones, with AUROC .971 against correct words (PARSeq .871): these are mostly multi-character errors far from the image. On printed data out-of-graph errors are harder (recall 71–81%, AUROC .920–.946); the 26 on IIIT5K are hardest (53.8%). Recall grows with edit distance (IAM 79.7%, 90.9%, 93.2% for 1, 2, $\geq$3 edits): multi-character errors are easier to verify but harder to repair. Adding up to ten TrOCR beam-search candidates raises IAM coverage only from 55.4% to 57.9% (FUNSD 82.8% to 85.1%), so coverage is limited by the recognisers, not by the graph construction.

Correlated Failures of both Recognisers: (Supplementary Table S44). On words both verifiers misread, LOCUS is still right on 62–84% when their readings differ. When both produce the *same* wrong reading and the claim equals it, the evidence is unanimous and wrong: LOCUS rejects only 0–11.5% of such errors (FUNSD 61 words, IAM 25, IIIT5K 11; 1–4% of all errors), and re-recognition rejects none. This is the failure mode to expect on unseen engines that resemble the verifiers; verifiers from different families reduce it, and a third independent verifier and audited monitoring are the practical safeguards.

Table 10. Errors whose truth is inside / outside the counterfactual graph ($k = 25$). in G: share of errors with the truth in $G(h)$ (%); recall: share of these errors rejected by LOCUS or by PARSeq agreement (%); AUROC: correct words versus in-graph or out-of-graph errors. Last columns: share of errors and recall of LOCUS at edit distance 1, 2 and 3 or more.

Dataset	errors	in G	recall (%)			AUROC			by edit distance 1/2/≥3	
			LOCUS in	LOCUS out	PARSeq out	LOCUS in	LOCUS out	PARSeq out	share (%)	LOCUS recall (%)
FUNSD	1474	82.8	92.5	71.3	74.4	0.985	0.920	0.934	50.7/29.2/20.1	81.0/95.6/99.0
CORD	941	94.9	97.4	81.2	85.4	0.996	0.939	0.945	49.9/21.5/28.6	94.0/98.0/100.0
SROIE	1880	92.6	97.0	77.7	94.2	0.995	0.946	0.635	47.0/20.8/32.2	91.3/98.5/99.8
IAM	1857	55.4	83.7	90.6	94.8	0.967	0.971	0.871	43.0/27.1/29.9	79.7/90.9/93.2
IIIT5K	869	97.0	99.3	53.8	53.8	0.997	0.795	0.852	30.0/18.9/51.1	93.5/99.4/100.0

6.7 Noisy Pseudo-labels and Confirmation Bias

Pretraining labels a word as correct when two engines agree (Tesseract and TrOCR-small; TrOCR-small and -base on IAM). Agreement precision, measured afterwards against training ground truth that is never used, ranges from 99.3% (IIIT5K) to 83.7% (IAM; Table 11), so shared engine errors could be learned as correct (confirmation bias). We tested three remedies, each pretrained and adapted exactly like LOCUS: dropping the 30% least confident pseudo-positives, weighting them by confidence rank, and requiring PARSeq to agree as well. All raise IAM precision (to 89.3%, 87.6% and 97.5%), and accuracy follows precision: IAM macro-F1 rises from .898 to .910, .905 and .917 (AUROC .967 to .973 with three engines). Where agreement is already 94–99% precise the remedies change macro-F1 by at most 0.005, except that the PARSeq veto lowers SROIE to .954, probably because PARSeq often misreads SROIE lines. Confirmation bias is thus real but small after k-shot adaptation, and confidence thresholding is a cheap default (it never cost more than 0.008 AUROC).

Table 11. Pseudo-label variants for label-free pretraining: precision of the pseudo-positive labels (%) | macro-F1 / AUROC of LOCUS after $k = 25$ adaptation.

Pseudo-labels	FUNSD	CORD	SROIE	IAM	IIIT5K
two-engine agreement (default)	94.5 | .942/.975	98.7 | .978/.993	99.0 | .974/.992	83.7 | .898/.967	99.3 | .971/.991
+ confidence threshold (drop lowest 30%)	96.4 | .940/.967	99.6 | .981/.993	99.2 | .970/.992	89.3 | .910/.971	99.8 | .971/.991
+ confidence weighting	96.1 | .939/.974	99.3 | .979/.993	99.3 | .973/.992	87.6 | .905/.969	99.6 | .969/.991
three-engine agreement	96.1 | .947/.967	99.5 | .977/.993	99.7 | .954/.989	97.5 | .917/.973	99.8 | .973/.992

7. Qualitative Analysis and Manual Audit

Qualitative examples, drawn by a seeded random choice inside rule-defined strata and showing LOCUS' failures as often as its successes, are given in Supplementary Figures S13 and S12.

Regimes: Supplementary Table S40 counts the two regimes that matter on the full test sets. Correct claims that PARSeq misreads are frequent on every dataset, and LOCUS accepts most of them (Supplementary Figure S12); this is the regime in which comparing hypotheses under two verifiers pays off. Character errors that PARSeq reproduces are rarer and LOCUS rejects only a small fraction of them (Supplementary Figure S5).

Manual Audit: To understand the errors behind these numbers, we inspected two random samples of up to 20 test words per dataset: (i) words on which LOCUS decides wrongly and (ii) incorrect claims that PARSeq agrees with. Each crop was labelled *genuine* (the ground truth matches the visible text), *label noise*, *crop problem* (text clipped or a neighbouring glyph inside the box) or *ambiguous* (Supplementary Table S41). Two conclusions follow. First, fewer than half of LOCUS' apparent mistakes are genuine verification errors on a correct label (42 of 100 sampled): 25 are label noise, 9 crop problems and 8 ambiguous, and on IIIT5K 16 of 20 are claims that count as correct only after the case-insensitive, punctuation-free normalisation of the protocol (*e.g.* `HoLLywooD` for an upper-case sign), which LOCUS, judging the exact appearance of the string, rejects; this protocol mismatch, rather than a visual error, explains most of LOCUS' gap to PARSeq agreement (which compares normalised strings) on IIIT5K. Second, the stratum "errors that PARSeq agrees with" consists mostly of cases where the claim matches what is visible: only 10 of 92 sampled words are genuine errors, against 46 label-noise and 21 crop problems, so low rejection rates there are not a weakness of either verifier. The genuine failures that remain fall into two identifiable modes: rejecting correct short or case-ambiguous claims when PARSeq reads another case (CORD: `Rp`, `x`, `Pop`), and accepting capitalisation errors in handwriting (IAM: `pinkie`, `market`, `church`). The audit was performed by a single annotator; the annotated sheets are released with the code.

8. Discussion and Limitations

When does Comparing Hypotheses Help?: LOCUS helps when its two recognisers *disagree in correctness*: on words only one verifier reads correctly, re-recognition with the other is right 26–62% of the time and LOCUS 64–99% (Supplementary Table S42). It gains most where recognisers are fallible and complementary (handwriting, camera receipts), least where one in-domain recogniser is nearly always right, and it does not require knowing in advance *which* recogniser to trust.

Limitations: All design decisions were made on FUNSD and CORD; SROIE, IAM and IIIT5K are held out.

1. Repair is bounded by graph coverage (about half of the handwriting errors); verification is not (Section 6.6).
2. Pretraining relies on engine agreement (84–99% precise); stricter consensus helps where it is imprecise (Section 6.7).
3. TrOCR's printed checkpoints were fine-tuned on SROIE and PARSeq was developed on IIIT5K; the re-recognition baselines use the same recognisers.

4. Message passing, the local-optimality term and render normalisation are individually redundant in the full model: the evidence, not the network, is the contribution.
5. Under protocols that normalise case and punctuation (IIIT5K) a claim such as `HoLLywooD` counts as correct but looks wrong.
6. All datasets are Latin-script; connected and complex scripts and historical documents need a shaping renderer, grapheme-level confusions and script-specific recognisers and are untested.
7. Evidence extraction runs at about 18 words/s; pruning and a cascade reach roughly 40 words/s, which suits the uncertain fraction of a production stream, not exhaustive industrial-scale verification.
8. When both recognisers make the same error and the claim agrees, LOCUS almost never detects it.
9. A new engine needs about 25 labelled words per class; label-free recalibration did not help.
10. The IAM macro-F1 gain holds for the ensemble but not every single model, and IAM is not writer-disjoint. Deployments should monitor accepted precision on audited samples.

9. Conclusion

We reframed OCR verification as asking whether a claim is the locally optimal explanation of its image among OCR-plausible counterfactual alternatives, and made the comparison visual with a render-normalised likelihood ratio under two frozen recognisers. LOCUS, a relation-aware graph network over counterfactual string graphs, is pretrained without human labels and adapts from a handful of labelled words. On a new benchmark of real OCR errors over five public datasets it outperforms all general-purpose verifiers in AUROC on every dataset and, against re-recognition with state-of-the-art recognisers, is the only verifier close to the best everywhere. Our analyses locate the benefit in the counterfactual evidence and label-free pretraining rather than in deep message passing: simple learners on the same evidence come within 0.015 macro-F1, so the graph network is a convenient rather than a necessary implementation. Future work will extend the confusion model and verifiers to non-Latin scripts and make the evidence aware of each dataset's comparison protocol.

Declarations

Funding

This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.

Ethical Statement

This study uses only publicly available datasets (FUNSD, CORD, SROIE, IAM and IIIT5K) under their respective licences and publicly released pretrained models. It involves no human participants, animals or personal data beyond what the public datasets contain, and therefore required no ethical approval.

Conflicts of Interest / Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data and Code Availability

The benchmark manifests, all code, raw result files and the annotated audit sheets, together with a step-by-step reproduction guide and the Supplementary Material, are available at an anonymised repository (link withheld for double-blind review).https://github.com/Shikhar2004-prog/LOCUS-OCR-Verification.

Prior Publication

This article extends a conference paper by the same authors (reference withheld for double-blind review).This article is an extended version of the conference paper by Saxena *et al.* [13], which verified OCR with character-level relational line graphs on one private dataset. The present article introduces a new segmentation-free model (counterfactual string graphs with render-normalised evidence and label-free pretraining), five public benchmarks with two OCR engines, new baselines and all analyses; no text, table or figure is reused from the conference version.

Author Contributions

Shikhar Dubey and Sanhita Saxena contributed equally to conceptualisation, methodology, software, experiments, analysis and writing.

Acknowledgement

The authors thank the creators of the FUNSD, CORD, SROIE, IAM and IIIT5K datasets and of the open-source TrOCR, PARSeq, Florence-2, Qwen2.5-VL and Tesseract models used in this work.

References

[1] Fu, Ling, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu et al. "Ocrbench v2: An Improved Benchmark for Evaluating Large Multimodal Models on Visual Text Localization and Reasoning." Advances in Neural Information Processing Systems 38 (2026).

[2] Zhang, Can, Ziheng Wu, Zhenghao Chen, Yufei Zhan, Yifan Li, Zhao Zhang, Xian Wang, and Minghui Qiu. "Seeing is Believing? Mitigating OCR Hallucinations in Multimodal Large Language Models." Advances in Neural Information Processing Systems 38 (2026): 74230-74248.

[3] Bautista, Darwin, and Rowel Atienza. "Scene Text Recognition with Permuted Autoregressive Sequence Models." In European conference on computer vision, Cham: Springer Nature Switzerland, 2022, 178-196.

[4] Li, Minghao, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. "Trocr: Transformer-based Optical Character Recognition with Pre-Trained Models." In Proceedings of the AAAI conference on artificial intelligence, vol. 37, no. 11, 2023, 13094-13102.

[5] Xiao, Bin, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. "Florence-2: Advancing a Unified Representation for a Variety of Vision Tasks." In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2024, 4818-4829.

[6] Qwen Team. (2025). Qwen2.5-VL-7B-Instruct [Vision-language model]. Alibaba Cloud. https://huggingface.co/Qwen/Qwen2.5-VL-7B-Instruct

[7] Corbiere, Charles, Nicolas Thome, Antoine Saporta, Tuan-Hung Vu, Matthieu Cord, and Patrick Perez. "Confidence Estimation via Auxiliary Models." IEEE Transactions on Pattern Analysis and Machine Intelligence 44, no. 10 (2021): 6043-6055.

[8] Kanerva, Jenna, Cassandra Ledins, Siiri Käpyaho, and Filip Ginter. "OCR Error Post-correction with LLMs in Historical Documents: No Free Lunches." In Proceedings of the Third Workshop on Resources and Representations for Under-Resourced Languages and Domains (RESOURCEFUL-2025), 2025, 38-47.

[9] Kim, Geewook, 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, Cham: Springer Nature Switzerland, 2022, 498-517.

[10] Poznanski, Jake, Kyle Lo, and Luca Soldaini. "The olmOCR Project: Building Fully Open OCR using VLMs." In Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), 2026, 626-635.

[11] Leng, Sicong, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. "Mitigating Object Hallucinations in Large Vision-Language Models Through Visual Contrastive Decoding." In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2024, 13872-13882.

[12] Dubey, Shikhar, Krish Mittal, Sourava Kumar Behera, Manikandan Ravikiran, Nitin Kumar, Saurabh Shigwan, and Rohit Saluja. "Multi-Feature Graph Convolution Network for Hindi OCR Verification." In Proceedings of the 1st Workshop on Benchmarks, Harmonization, Annotation, and Standardization for Human-Centric AI in Indian Languages (BHASHA 2025), 2025, 1-10.

[13] Saxena, S., Tripathi, V., & Abbas, S. (2026). Reading between the characters: Label-efficient OCR verification with graph neural networks. In Proceedings of NICCAI 2026, Hyderabad, India (to appear).

[14] Brody, Shaked, Uri Alon, and Eran Yahav. "How Attentive are Graph Attention Networks?." arXiv preprint arXiv:2105.14491 (2021).

[15] Arazo, Eric, Diego Ortego, Paul Albert, Noel E. O'Connor, and Kevin McGuinness. "Pseudo-labeling and Confirmation Bias in Deep Semi-Supervised Learning." In 2020 International joint conference on neural networks (IJCNN), IEEE, 2020, 1-8.

[16] Sohn, Kihyuk, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. "Fixmatch: Simplifying Semi-Supervised Learning with Consistency and Confidence." Advances in neural information processing systems 33 (2020): 596-608.

[17] Minderer, Matthias, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. "Revisiting the Calibration Of Modern Neural Networks." Advances in neural information processing systems 34 (2021): 15682-15694.

[18] Alexandari, Amr, Anshul Kundaje, and Avanti Shrikumar. "Maximum Likelihood with Bias-Corrected Calibration is Hard-to-Beat at Label Shift Adaptation." In International conference on machine learning, PMLR, 2020, 222-232.

[19] Angelopoulos, Anastasios N., and Stephen Bates. "Conformal Prediction: A Gentle Introduction." Foundations and Trends in Machine Learning 16, no. 4 (2023): 494-591.

[20] JJaume, Guillaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. "Funsd: A Dataset for Form Understanding in Noisy Scanned Documents." In 2019 international conference on document analysis and recognition workshops (ICDARW), vol. 2, IEEE, 2019, 1-6.

[21] Park, Seunghyun, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. "Cord: a Consolidated Receipt Dataset for Post-ocr Parsing." In Workshop on document intelligence at NeurIPS, vol. 2019, no. 1, p. 5. 2019.

[22] Huang, Zheng, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and C. V. Jawahar. "Icdar2019 Competition on Scanned Receipt Ocr and Information Extraction." In 2019 International Conference on Document Analysis and Recognition (ICDAR), IEEE, 2019, 1516-1520.

[23] Tesseract OCR contributors. (2021). Tesseract OCR (Version 5) [Computer software]. ht tps://github.com/tesseract-ocr/tesseract

Appendix:

Supplementary Material

This document supplements the article *LOCUS: Verifying OCR by Local Optimality over Counterfactual String Graphs*. It gives the implementation details needed for reproduction and the complete experimental record behind the summaries of the main article; references without the prefix S (e.g. "Table 3") point to the main article, and numbered citations such as [6] refer to its reference list. All tables and figures are generated by scripts from the raw result files; no number was entered by hand.

S1. Benchmark Construction

Crops: FUNSD words are cropped from the page with a 2-pixel margin after rescaling the LayoutLM-normalised boxes ($[0, 1000]^2$) of the public Hugging Face release to pixel coordinates; cropping with the raw values yields misaligned crops for almost every word (the Tesseract word error rate is 99.8% without the rescaling and 36.9% with it). CORD words are cropped from the axis-aligned bounding box of each word quadrilateral; the official train and validation receipts form the training pool. SROIE, IAM and IIIT5K are distributed as crops. Crops narrower than 6 or shorter than 8 pixels, and empty transcriptions, are discarded.

Engines: Tesseract 5 (LSTM engine, `--psm 7`, crops shorter than 32 px are bicubically upscaled to 32 px) outputs the space-joined words it detects and their mean confidence. TrOCR-small (`microsoft/trocr-small-printed` or `-handwritten`) decodes greedily; its confidence is the mean token probability.

Labels: A claim is CORRECT iff it equals the ground truth after whitespace normalisation (case-insensitive for SROIE, whose ground truth is upper-case; case-insensitive and restricted to alphanumeric characters for IIIT5K, following its standard protocol). No other normalisation is applied, so case and punctuation errors count as errors, as they do for any downstream consumer that copies the field verbatim.

Caveats: TrOCR's printed checkpoints were fine-tuned on SROIE receipts; the verifier V_1 is therefore in-domain on SROIE, and TrOCR-small makes only 3–6% errors there, which is why SROIE uses Tesseract claims. Tesseract is degenerate on handwriting (93.9% word error on IAM), which is why IAM uses TrOCR-small claims.

S2. Counterfactual Neighbourhood Generation

Confusion Groups: Substitutions are drawn from the groups `0Oo`, `QDC`, `c`, `Il1`, `|!ij`, `5Ss$`, `8B&3`, `2Zz`, `6Gb`, `9gq`, `7T/`, `uvnUV`, `wW`, `.,:;'` `"`, `-_~=`, `(){}[]`, `aeo`, `hbk`, `rnm`, `cC(e`, `xX%`, `yY`, `pP`, `fft`; at most two alternatives per character are used.

Algorithm: Given claim h and verifier readings g_1 (TrOCR) and g_2 (PARSeq):

1. add g_2 and g_1 (if different from h);
2. for each reading, add every single edit of h suggested by its Levenshtein alignment to the reading: substitution of an aligned mismatching character, deletion of an unaligned character of h, insertion of an unaligned character of the reading;
3. for each position of h (up to 32): a case flip, up to two confusion substitutions, and a deletion;
4. deduplicate, keep insertion order, and stop at $M = 28$ neighbours.

No step uses the ground truth. The share of real errors whose truth is a node of the graph is reported in Table S33.

Rendering: $R(v)$ draws v in Liberation Sans (metrically equivalent to Arial) at $0.75\times$ the crop height (clamped to $[32, 96]$ px), black on white, with a quarter-height margin. The same rendering is fed to both verifiers. PARSeq scores the string restricted to its 94-character charset and truncated to its maximum label length of 25.

S3. Label-free Pretraining Data

From the unlabeled training pool we keep images on which two independent engines produce the identical string (Tesseract and TrOCR-small; TrOCR-small and TrOCR-base on IAM), shuffle them with a fixed seed and keep at most 3,000. Each yields a pseudo-correct graph (the agreed string) and, if the corruption changes the string, a pseudo-incorrect graph built around one or two random OCR-style edits: confusable substitution (45%), case flip (15%), deletion (20%) or insertion of a confusable character (20%). The repair target of a corrupted graph is the node holding the agreed string (when present). Agreement is a proxy, not ground truth; its precision on the training pool is reported in Table S1.

Table S1. Label-free pretraining sets. Precision of engine agreement is measured post hoc against the ground truth of the training pool (never used for training).

Dataset	Agreeing images	Graphs	Precision of agreement	Truth in graph (corruptions)
FUNSD	2,561	5,085	94.5%	99.6%
CORD	3,000	5,959	98.7%	99.8%
SROIE	2,663	5,301	99.0%	100.0%
IAM	3,000	5,950	83.7%	100.0%
IIIT5K	860	1,711	99.3%	99.5%

S4. Hyperparameters and Baseline Protocols

Table S2. Hyperparameters (identical for all datasets and all k).

Node features	2×11 evidence + 7 claim deltas (tanh-squashed) + 8 edit-kind + 1 position + 6 agreement (claim node)
Edge features	8 edit kinds + sibling + path + self (one-hot)
GNN	3 GATv2 layers, hidden 64, 4 heads, residual + LayerNorm, dropout 0.2
Readout	claim state, attention pool, max pool over rivals, $\beta \log P_{\text{opt}}$ (β learned)
Pretraining	15 epochs, AdamW lr 10^{-3}, wd 10^{-4}, batch 64, $\lambda = 0.5$
Few-shot fine-tuning	300 steps of the whole pretrained GNN, AdamW lr $3 \cdot 10^{-4}$, batch = half the support set, class-balanced BCE
Inference	average of the five networks fine-tuned on the five support sets (seeds 0–4); threshold 0.5
Ablation-only heads	CV-selected prior/anchored head and linear probe (Table 4 only; not part of LOCUS)
Full supervision	60 epochs, early stopping (patience 8) on a 15% split of the training pool
Seeds	0–4 (support sampling, initialisation, pretraining)

Thresholds: Engine confidence and the ρ-margin rule are thresholded at the value maximising macro-F1 on the k-shot support (seed-specific). Agreement baselines are binary (accept iff equal). The graded PARSeq score is $+c$ if PARSeq's reading equals the claim and

$-c$ otherwise, where c is PARSeq's sequence confidence (product of character probabilities); it is thresholded at 0, which coincides with the binary decision.

VLM Judge Prompt: *Does the image show exactly the text "{claim}" (same characters, case and punctuation)? Answer yes or no.* The image is passed at its native resolution (upscaled to a height of 56 px if smaller; Qwen2.5-VL pixel budget 32–256 visual tokens). The score is the renormalised probability of `yes`/`Yes` versus `no`/`No` as the first generated token.

ConfidNet-style Learned Confidence: A one-hidden-layer MLP (32 units, ℓ_2 10^{-2}) on the engine confidence and five likelihood statistics of the claim under TrOCR-base, trained on the support set.

S5. Additional Analyses and Qualitative Examples

Error Types

Table S3 breaks down the detection rate (recall of truly incorrect claims) by the edit that separates the claim from the truth. Operating points: LOCUS at $P = 0.5$, PARSeq-graded at its natural threshold, the VLM at 0.5, and engine confidence at its *test-optimal* threshold (favourable to the baseline).

Table S3. Detection rate (%) by error type; number of errors of that type in parentheses ($k = 25$).

Dataset	Method	substitution	insertion	deletion	case only	multi-edit
FUNSD	LOCUS	87.7 (236)	79.7 (390)	71.8 (78)	77.2 (57)	97.1 (713)
	PARSeq re-OCR (graded)	86.0 (236)	80.3 (390)	70.5 (78)	78.9 (57)	98.2 (713)
	TrOCR-base re-OCR	93.2 (236)	85.1 (390)	82.1 (78)	96.5 (57)	98.5 (713)
	Qwen2.5-VL-7B judge	62.3 (236)	37.7 (390)	67.9 (78)	26.3 (57)	79.2 (713)
	Engine confidence	74.2 (236)	76.4 (390)	48.7 (78)	71.9 (57)	89.3 (713)
CORD	LOCUS	95.3 (342)	93.2 (73)	92.5 (40)	80.0 (20)	99.1 (466)
	PARSeq re-OCR (graded)	95.6 (342)	98.6 (73)	95.0 (40)	100.0 (20)	100.0 (466)
	TrOCR-base re-OCR	98.0 (342)	100.0 (73)	95.0 (40)	90.0 (20)	100.0 (466)
	Qwen2.5-VL-7B judge	74.3 (342)	20.5 (73)	87.5 (40)	10.0 (20)	94.2 (466)
	Engine confidence	64.6 (342)	53.4 (73)	92.5 (40)	55.0 (20)	92.9 (466)
SROIE	LOCUS	97.5 (641)	76.6 (137)	72.4 (105)	–	99.3 (997)
	PARSeq re-OCR (graded)	98.9 (641)	100.0 (137)	79.0 (105)	–	99.8 (997)
	TrOCR-base re-OCR	99.5 (641)	82.5 (137)	79.0 (105)	–	99.4 (997)
	Qwen2.5-VL-7B judge	68.2 (641)	29.2 (137)	37.1 (105)	–	90.4 (997)
	Engine confidence	75.5 (641)	75.9 (137)	44.8 (105)	–	92.8 (997)
IAM	LOCUS	93.5 (505)	76.8 (95)	57.7 (26)	43.2 (185)	92.8 (1046)
	PARSeq re-OCR (graded)	95.6 (505)	98.9 (95)	57.7 (26)	94.6 (185)	98.1 (1046)
	TrOCR-base re-OCR	88.9 (505)	85.3 (95)	76.9 (26)	29.2 (185)	95.9 (1046)
	Qwen2.5-VL-7B judge	80.0 (505)	61.1 (95)	61.5 (26)	57.8 (185)	86.3 (1046)
	Engine confidence	82.8 (505)	68.4 (95)	69.2 (26)	27.0 (185)	89.5 (1046)
IIIT5K	LOCUS	92.4 (172)	96.4 (28)	95.1 (61)	–	99.8 (608)
	PARSeq re-OCR (graded)	94.2 (172)	100.0 (28)	95.1 (61)	–	99.8 (608)
	TrOCR-base re-OCR	93.6 (172)	100.0 (28)	93.4 (61)	–	99.8 (608)
	Qwen2.5-VL-7B judge	89.0 (172)	96.4 (28)	93.4 (61)	–	99.7 (608)
	Engine confidence	50.6 (172)	60.7 (28)	65.6 (61)	–	94.1 (608)

Claim Length

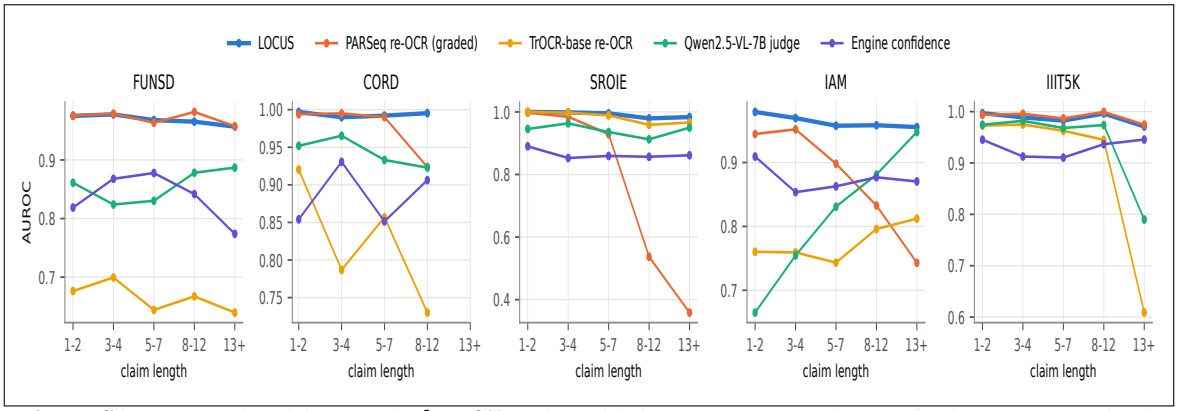

Figure S1. AUROC by claim length ($k = 25$). Bins with fewer than 30 words or a single class are omitted.

Macro-F1 by Claim Length: (Table S4). Accuracy decreases with claim length on every dataset, most on SROIE lines and IAM words, where long claims also hit the cap of $M = 28$ rivals (Section 5.6).

Train/test Duplicates: A pixel-level hash comparison of all training and test crops finds no shared image on FUNSD, SROIE, IAM and IIIT5K. On CORD, 79 test crops are pixel-identical to training crops because some receipts occur in both official splits; removing them leaves LOCUS' macro-F1 (.978) and AUROC (.993) unchanged.

Table S4. LOCUS macro-F1 by claim length in characters ($k = 25$; number of test words in parentheses; bins with fewer than 30 words or one class omitted).

Dataset	1–3	4–7	8–15	≥16
FUNSD	0.951 (1358)	0.935 (1731)	0.940 (903)	–
CORD	0.984 (746)	0.975 (1367)	0.975 (237)	–
SROIE	0.992 (612)	0.983 (1331)	0.963 (975)	0.959 (1082)
IAM	0.933 (1789)	0.881 (1599)	0.869 (595)	–
IIIT5K	0.979 (1117)	0.973 (1453)	0.971 (404)	–

Table S5. Seed-level comparison: each of the five $k = 25$ models (support seeds 0–4) separately against the strongest baseline; difference in AUROC and in macro-F1.

Dataset	baseline	ΔAUROC, seeds 0–4					Δmacro-F1, seeds 0–4				
		0	1	2	3	4	0	1	2	3	4
FUNSD	PARSeq (graded)	-0.006	-0.012	-0.009	-0.015	-0.008	-0.028	-0.012	-0.018	-0.026	-0.021
CORD	PARSeq (graded)	+0.006	+0.007	+0.007	-0.006	+0.007	-0.019	+0.006	+0.007	-0.044	+0.008
SROIE	TrOCR-base	+0.011	+0.014	+0.011	+0.011	+0.013	-0.023	-0.011	-0.028	-0.016	-0.026
IAM	PARSeq (graded)	+0.031	+0.046	+0.042	+0.044	+0.030	+0.000	+0.015	-0.001	-0.003	-0.016
IIIT5K	PARSeq (graded)	-0.005	-0.008	-0.006	-0.010	-0.006	-0.018	-0.059	-0.024	-0.041	-0.023

Calibration

Table S6. Expected calibration error (15 bins) of the correctness probability ($k = 25$; PARSeq-graded mapped to $[0, 1]$ as $(s + 1)/2$). Lower is better.

Dataset	LOCUS	PARSeq graded	Qwen2.5-VL-7B	Engine conf.
FUNSD	0.024	0.039	0.019	0.136
CORD	0.025	0.018	0.045	0.114
SROIE	0.020	0.197	0.057	0.213
IAM	0.030	0.058	0.110	0.289
IIIT5K	0.027	0.009	0.136	0.042

AUROC and AURC Learning Curves

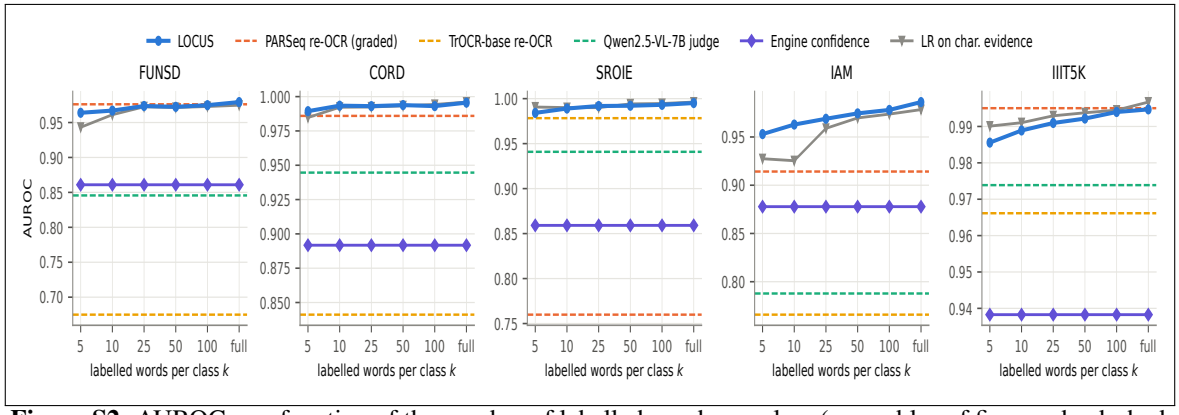

Figure S2. AUROC as a function of the number of labelled words per class (ensembles of five seeds; dashed: label-free baselines).

Cross-dataset Transfer (AUROC)

Table S7. Cross-dataset transfer, AUROC (rows: training dataset, fully supervised; columns: test dataset; diagonal: in-domain).

Train \ Test	FUNSD	CORD	SROIE	IAM	IIIT5K
FUNSD	.979	.995	.979	.968	.993
CORD	.968	.996	.987	.954	.986
SROIE	.941	.992	.996	.892	.992
IAM	.962	.989	.952	.985	.991
IIIT5K	.961	.993	.975	.963	.995

Ablations: AUROC and Repair

Table S8. Ablations at $k = 25$, AUROC (ensemble of five seeds). Last column: mean change w.r.t. the full model.

Variant	FUNSD	CORD	SROIE	IAM	IIIT5K	Δ
LOCUS (full)	.973	.993	.992	.968	.991	
+ CV-selected prior/anchored head	.974	.993	.990	.968	.992	-0.000
+ linear probe on embedding	.974	.993	.992	.969	.993	+0.001
w/o verifier-agreement node features	.974	.993	.991	.968	.987	-0.001

w/o edit-graph GNN (char evidence LR)	.973	.992	.990	.959	.993	-0.002
w/o label-free pretraining	.969	.992	.990	.970	.990	-0.002
w/o local-optimality term	.972	.993	.992	.966	.990	-0.001
w/o repair loss	.971	.993	.992	.965	.990	-0.001
w/o render-normalisation	.973	.993	.992	.967	.992	-0.000
w/o latent render distance	.973	.993	.992	.969	.990	-0.000
likelihood only (no render)	.974	.992	.991	.969	.992	+0.000
TrOCR verifier only	.964	.991	.992	.956	.991	-0.005
PARSeq verifier only	.974	.993	.990	.965	.992	-0.001
star edges only	.971	.993	.992	.969	.991	-0.001
no message passing	.975	.994	.992	.967	.991	+0.000
GCN backbone	.970	.993	.991	.966	.990	-0.002
GraphSAGE backbone	.973	.993	.992	.970	.990	-0.000
GIN backbone	.973	.991	.989	.957	.990	-0.004

Sensitivity

Table S9. Sensitivity of LOCUS ($k = 25$, ensemble of five seeds) to the size of the label-free pretraining set, the number of fine-tuning steps, and the support-selection strategy (label-free k-means clusters the unlabeled pool in the pretrained embedding space and labels the $2k$ medoids).

Factor	**Value**	**FUNSD**		**CORD**		**IAM**	
		F1	**AUROC**	**F1**	**AUROC**	**F1**	**AUROC**
pretrain fraction	0.1	.943	.967	.977	.993	.902	.968
pretrain fraction	0.25	.944	.968	.978	.994	.908	.970
pretrain fraction	0.5	.941	.967	.979	.994	.904	.970
pretrain fraction	1.0	.940	.974	.980	.993	.900	.967
fine-tuning steps	100	.942	.976	.975	.993	.884	.960
fine-tuning steps	300	.940	.974	.980	.993	.900	.967
fine-tuning steps	1000	.938	.974	.979	.993	.899	.967
support selection	random	.940	.974	.980	.993	.900	.967
support selection	label-free k-means	.935	.964	.980	.994	.907	.964

Main Comparison at Other Label Budgets and Metrics

Table S10. Macro-F1 at $k = 10$ (protocol of Table 2).

Method	**FUNSD**	**CORD**	**SROIE**	**IAM**	**IIIT5K**	**Avg.**
Baselines						
Engine MSP confidence	.722	.775	.749	.782	.817	.769

ConfidNet-style learned conf.[7]	.702	.813	.785	.783	.915	.800
Two-engine voting	.609	.772	.976	.759	.862	.796
Re-OCR: TrOCR-base[4]	.605	.811	.979	.758	.952	.821
Re-OCR: Florence-2[5]	.647	.615	.478	.369	.823	.586
Re-OCR: PARSeq[3]	.946	.973	.742	.879	.993	.907
Re-OCR: PARSeq, graded	.946	.973	.742	.879	.993	.907
VLM judge: Qwen2.5-VL-7B[6]	.760	.891	.866	.700	.888	.821
VLM judge, oracle threshold†	.762	.902	.883	.719	.928	.839
Best baseline per dataset, chosen in hindsight‡	.946	.973	.979	.879	.993	.954
Strong learned baselines on our evidence						
LR on flattened graph evidence	.931	.957	.939	.881	.974	.937
LR on character evidence	.915	.970	.949	.839	.965	.927
GBDT on all evidence	.885	.960	.967	.882	.947	.928
DeepSets over rivals (no edges)	.931	.978	.930	.896	.964	.940
LOCUS-GNN from scratch	.920	.969	.913	.877	.961	.928
Ours						
LOCUS zero-shot (no labels)	.918	.904	.798	.825	.968	.883
LOCUS (k-shot)	.936	.980	.954	.888	.975	.947

Table S11. Macro-F1 at $k = 100$.

Method	**FUNSD**	**CORD**	**SROIE**	**IAM**	**IIIT5K**	**Avg.**
Baselines						
Engine MSP confidence	.771	.799	.777	.801	.790	.788
ConfidNet-style learned conf.[7]	.830	.905	.873	.832	.924	.873
Two-engine voting	.609	.772	.976	.759	.862	.796
Re-OCR: TrOCR-base[4]	.605	.811	.979	.758	.952	.821
Re-OCR: Florence-2[5]	.647	.615	.478	.369	.823	.586
Re-OCR: PARSeq[3]	.946	.973	.742	.879	.993	.907
Re-OCR: PARSeq, graded	.946	.973	.742	.879	.993	.907

VLM judge: Qwen2.5-VL-7B[6]	.760	.891	.866	.700	.888	.821
VLM judge, oracle threshold†	.762	.902	.883	.719	.928	.839
Best baseline per dataset, chosen in hindsight‡	.946	.973	.979	.879	.993	.954
Strong learned baselines on our evidence						
LR on flattened graph evidence	.945	.980	.978	.918	.970	.958
LR on character evidence	.940	.979	.977	.920	.967	.956
GBDT on all evidence	.944	.980	.979	.923	.983	.962
DeepSets over rivals (no edges)	.942	.980	.976	.926	.968	.958
LOCUS-GNN from scratch	.938	.978	.977	.927	.973	.959
Ours						
LOCUS zero-shot (no labels)	.918	.904	.798	.825	.968	.883
LOCUS (k-shot)	.946	.980	.974	.927	.973	.960

Table S12. Macro-F1 with full supervision (85% of the training pool; 15% for early stopping)

Method	**FUNSD**	**CORD**	**SROIE**	**IAM**	**IIIT5K**	**Avg.**
Baselines						
Engine MSP confidence	.787	.805	.790	.807	.854	.809
ConfidNet-style learned conf.[7]	.856	.934	.912	.859	.957	.904
Two-engine voting	.609	.772	.976	.759	.862	.796
Re-OCR: TrOCR-base[4]	.605	.811	.979	.758	.952	.821
Re-OCR: Florence-2[5]	.647	.615	.478	.369	.823	.586
Re-OCR: PARSeq[3]	.946	.973	.742	.879	.993	.907
Re-OCR: PARSeq, graded	.946	.973	.742	.879	.993	.907
VLM judge: Qwen2.5-VL-7B[6]	.760	.891	.866	.700	.888	.821
VLM judge, oracle threshold†	.762	.902	.883	.719	.928	.839
Best baseline per dataset, chosen in hindsight‡	.946	.973	.979	.879	.993	.954
Strong learned baselines on our evidence						
LR on flattened graph evidence	.949	.977	.978	.934	.976	.963

LR on character evidence	.948	.980	.979	.928	.976	.962
GBDT on all evidence	.951	.980	.978	.944	.989	.968
DeepSets over rivals (no edges)	.949	.983	.978	.947	.982	.968
LOCUS-GNN from scratch	.951	.983	.979	.950	.987	.970
Ours						
LOCUS zero-shot (no labels)	.918	.904	.798	.825	.968	.883
LOCUS (k-shot)	.950	.981	.979	.945	.985	.968

Table S13. AUROC at $k = 10$.

Method	**FUNSD**	**CORD**	**SROIE**	**IAM**	**IIIT5K**	**Avg.**
Baselines						
Engine MSP confidence	.861	.892	.859	.878	.938	.886
ConfidNet-style learned conf.[7]	.763	.907	.867	.849	.981	.873
Two-engine voting	.680	.807	.976	.767	.905	.827
Re-OCR: TrOCR-base[4]	.675	.841	.978	.766	.966	.845
Re-OCR: Florence-2[5]	.715	.685	.578	.522	.879	.676
Re-OCR: PARSeq[3]	.939	.975	.763	.885	.991	.911
Re-OCR: PARSeq, graded	.976	.986	.760	.914	.995	.926
VLM judge: Qwen2.5-VL-7B[6]	.846	.945	.941	.788	.974	.899
VLM judge, oracle threshold†	.846	.945	.941	.788	.974	.899
Best baseline per dataset, chosen in hindsight‡	.976	.986	.978	.914	.995	.970
Strong learned baselines on our evidence						
LR on flattened graph evidence	.972	.995	.991	.961	.991	.982
LR on character evidence	.961	.992	.990	.925	.991	.972
GBDT on all evidence	.952	.988	.986	.948	.982	.971
DeepSets over rivals (no edges)	.960	.995	.985	.960	.988	.978
LOCUS-GNN from scratch	.955	.992	.977	.957	.987	.974
Ours						
LOCUS zero-shot (no labels)	.949	.977	.966	.909	.983	.957
LOCUS (k-shot)	.967	.993	.989	.963	.989	.980

Table S14. AUROC at $k = 100$.

Method	FUNSD	CORD	SROIE	IAM	IIIT5K	Avg.
Baselines						
Engine MSP confidence	.861	.892	.859	.878	.938	.886
ConfidNet-style learned conf.[7]	.907	.964	.948	.894	.985	.940
Two-engine voting	.680	.807	.976	.767	.905	.827
Re-OCR: TrOCR-base[4]	.675	.841	.978	.766	.966	.845
Re-OCR: Florence-2[5]	.715	.685	.578	.522	.879	.676
Re-OCR: PARSeq[3]	.939	.975	.763	.885	.991	.911
Re-OCR: PARSeq, graded	.976	.986	.760	.914	.995	.926
VLM judge: Qwen2.5-VL-7B[6]	.846	.945	.941	.788	.974	.899
VLM judge, oracle threshold†	.846	.945	.941	.788	.974	.899
Best baseline per dataset, chosen in hindsight‡	.976	.986	.978	.914	.995	.970
Strong learned baselines on our evidence						
LR on flattened graph evidence	.975	.996	.994	.976	.993	.987
LR on character evidence	.973	.994	.995	.974	.994	.986
GBDT on all evidence	.975	.995	.995	.976	.996	.987
DeepSets over rivals (no edges)	.972	.995	.993	.979	.995	.987
LOCUS-GNN from scratch	.970	.994	.993	.977	.995	.986
Ours						
LOCUS zero-shot (no labels)	.949	.977	.966	.909	.983	.957
LOCUS (k-shot)	.975	.993	.993	.978	.994	.987

Table S15. AURC at $k = 25$ (lower is better; best = lowest).

Method	FUNSD	CORD	SROIE	IAM	IIIT5K	Avg.
Baselines						
Engine MSP confidence	.144	.140	.211	.203	.069	.153
ConfidNet-style learned conf.[7]	.164	.124	.184	.226	.054	.150
Two-engine voting	.220	.170	.152	.272	.076	.178
Re-OCR: TrOCR-base[4]	.229	.154	.151	.273	.058	.173
Re-OCR: Florence-2[5]	.198	.233	.377	.424	.083	.263

Re-OCR: PARSeq[3]	.125	.109	.246	.186	.050	.143
Re-OCR: PARSeq, graded	.090	.099	.242	.168	.049	.130
VLM judge: Qwen2.5-VL-7B[6]	.153	.127	.172	.250	.056	.151
VLM judge, oracle threshold†	.153	.127	.172	.250	.056	.151
Best baseline per dataset, chosen in hindsight‡	.090	.099	.151	.168	.049	.111
Strong learned baselines on our evidence						
LR on flattened graph evidence	.092	.096	.139	.146	.052	.105
LR on character evidence	.094	.098	.139	.152	.050	.107
GBDT on all evidence	.098	.100	.138	.150	.052	.108
DeepSets over rivals (no edges)	.094	.097	.138	.147	.052	.106
LOCUS-GNN from scratch	.095	.098	.139	.149	.052	.107
Ours						
LOCUS zero-shot (no labels)	.110	.104	.152	.180	.054	.120
LOCUS (k-shot)	.092	.097	.139	.149	.052	.106

Table S16. Coverage at 95% accepted precision at $k = 25$ (higher is better)

Method	**FUNSD**	**CORD**	**SROIE**	**IAM**	**IIIT5K**	**Avg.**
Baselines						
Engine MSP confidence	.346	.408	.091	.091	.631	.313
ConfidNet-style learned conf.[7]	.053	.465	.233	.007	.726	.297
Two-engine voting	.021	.409	.547	.001	.635	.322
Re-OCR: TrOCR-base[4]	.000	.447	.550	.001	.715	.343
Re-OCR: Florence-2[5]	.311	.249	.092	.001	.595	.249
Re-OCR: PARSeq[3]	.006	.615	.305	.455	.746	.426
Re-OCR: PARSeq, graded	.646	.614	.304	.454	.747	.553
VLM judge: Qwen2.5-VL-7B[6]	.147	.351	.238	.052	.716	.301
VLM judge, oracle threshold†	.147	.351	.238	.052	.716	.301
Best baseline per dataset, chosen in hindsight‡	.646	.615	.550	.455	.747	.603
Strong learned baselines on our evidence						

LR on flattened graph evidence	.645	.625	.546	.483	.745	.609
LR on character evidence	.631	.624	.545	.446	.745	.598
GBDT on all evidence	.625	.621	.548	.467	.746	.601
DeepSets over rivals (no edges)	.639	.627	.552	.497	.747	.612
LOCUS-GNN from scratch	.630	.624	.550	.482	.747	.607
Ours						
LOCUS zero-shot (no labels)	.270	.551	.401	.346	.739	.461
LOCUS (k-shot)	.628	.626	.553	.486	.744	.607

Composition of Real OCR Errors

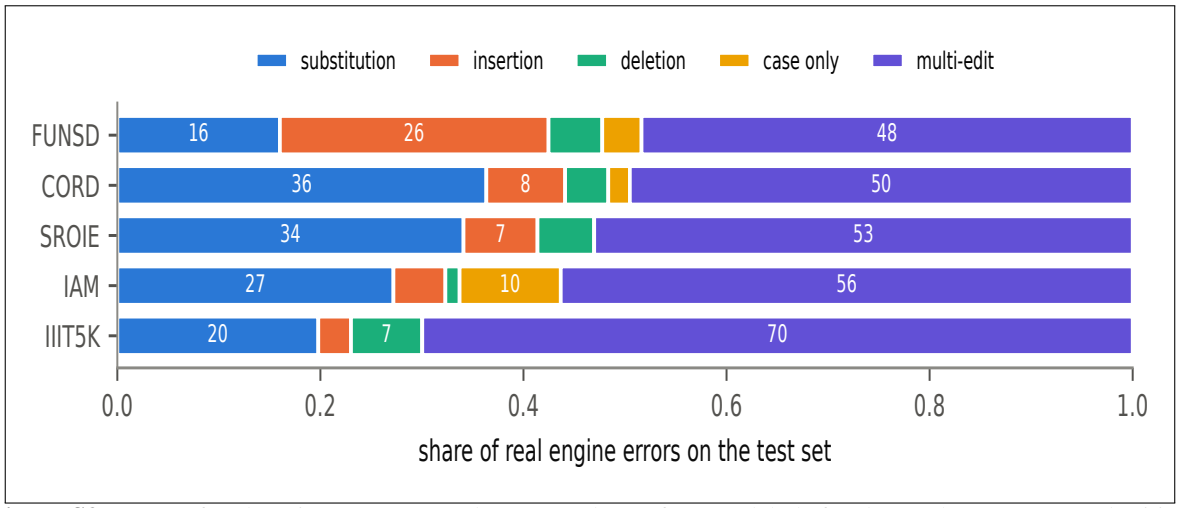

Figure S3. Types of real engine errors on each test set (share of errors; labels for shares above 7%). Handwriting and receipt lines are dominated by multi-edit errors, which bounds what a single-edit neighbourhood can repair (Table S33).

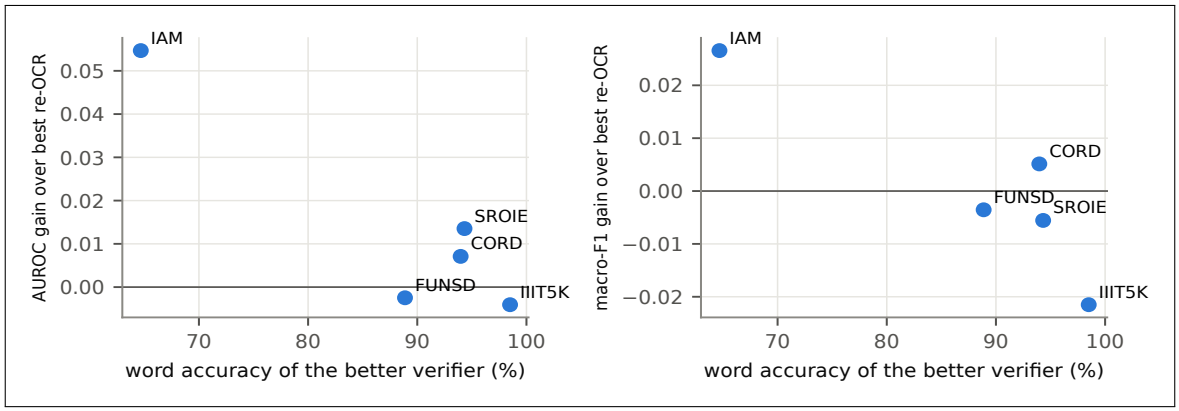

Figure S4. Gain of LOCUS over the best re-recognition baseline against the word accuracy of the better verifier (one point per dataset).

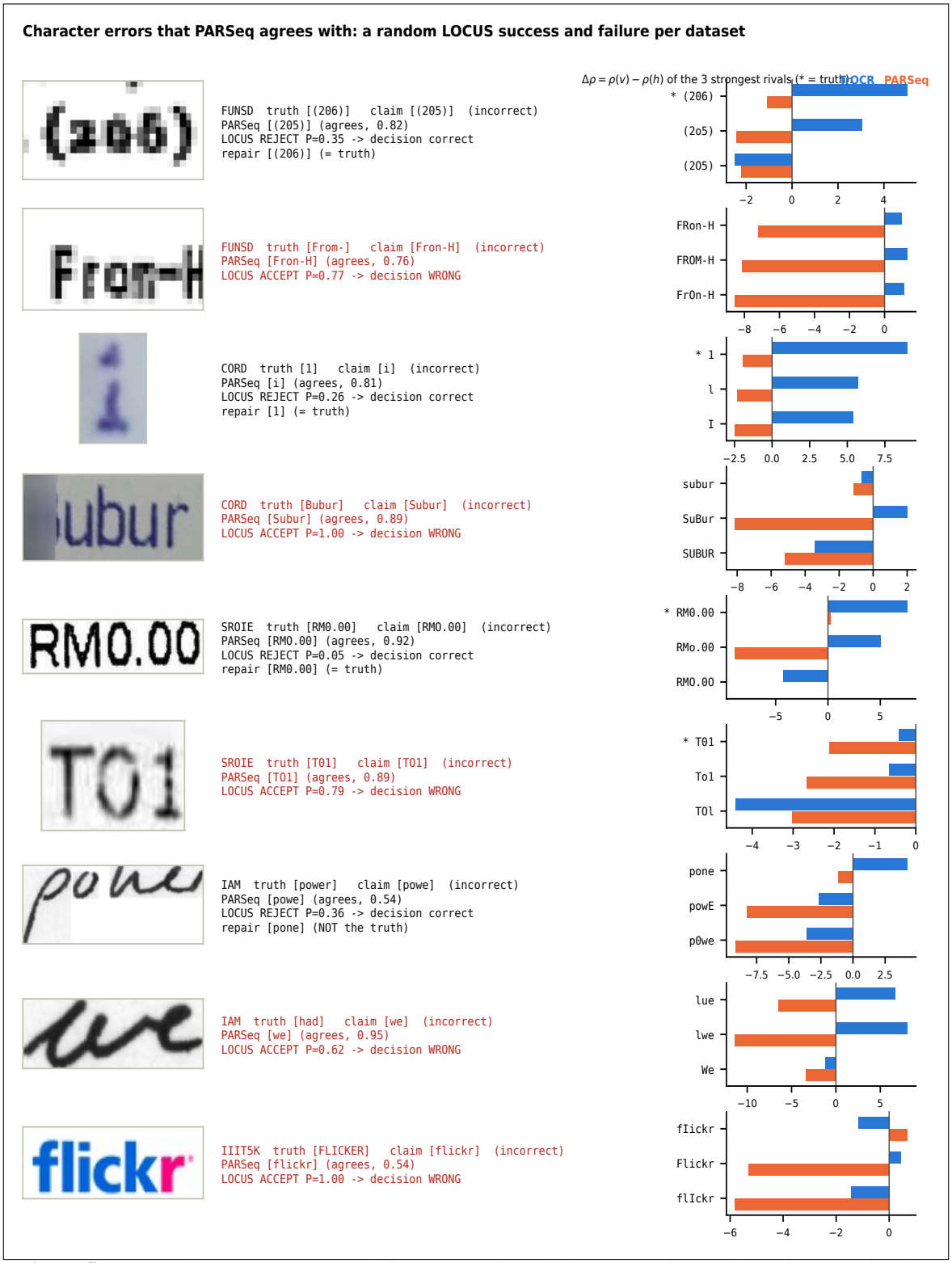

Figure S5. Errors that PARSeq re-recognition accepts (it reads the same string as the engine), restricted to genuine character errors (case-only and punctuation-only differences excluded): for each dataset one random LOCUS success and one random failure. LOCUS rejects only a small fraction of this stratum (Table S40); the manual audit (Table S41) shows why: most of these "errors" are annotation or cropping problems, i.e. the claim matches what is visible.

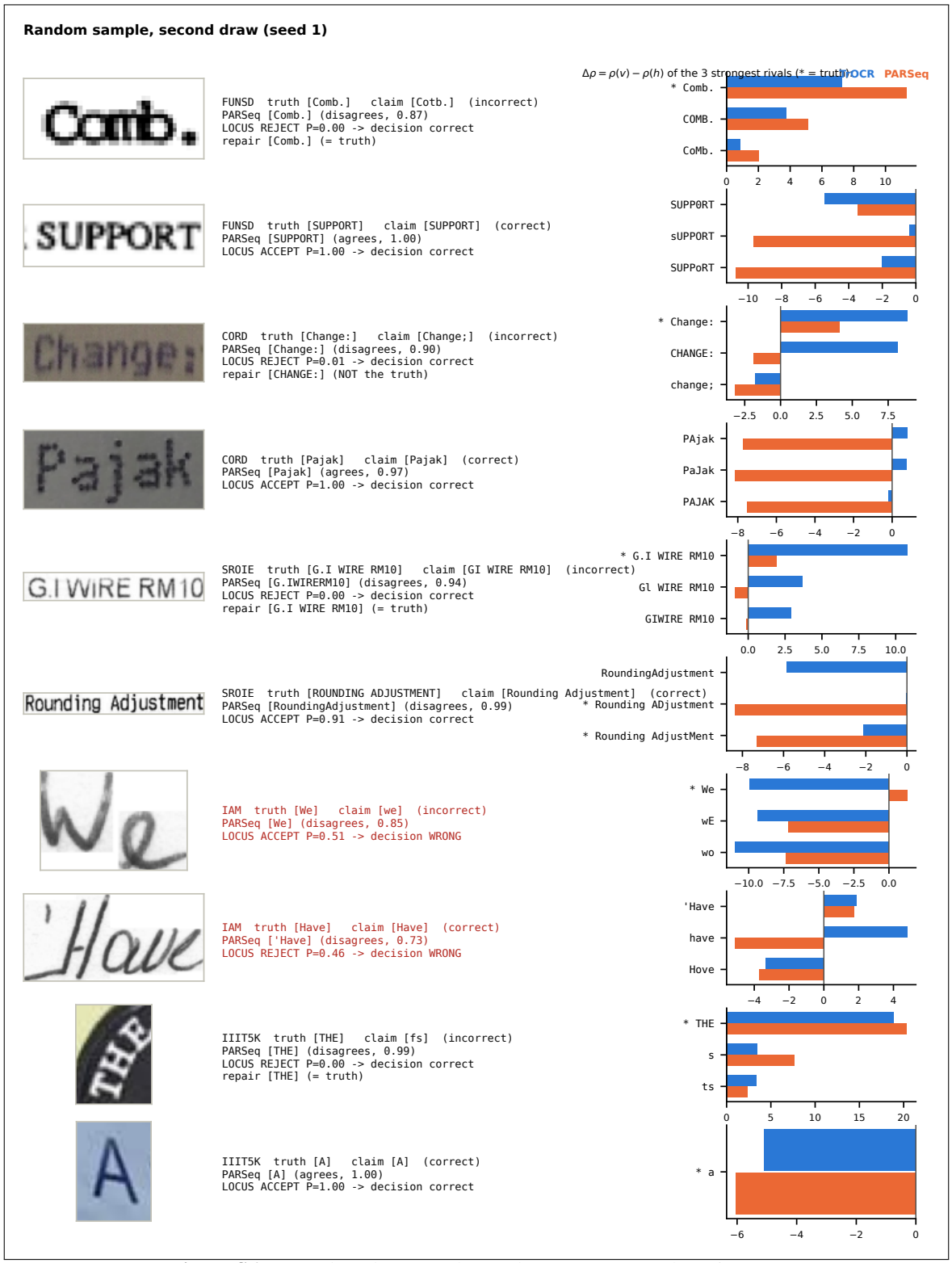

Figure S6. Second random sample (seed 1), same protocol as Fig. S13.

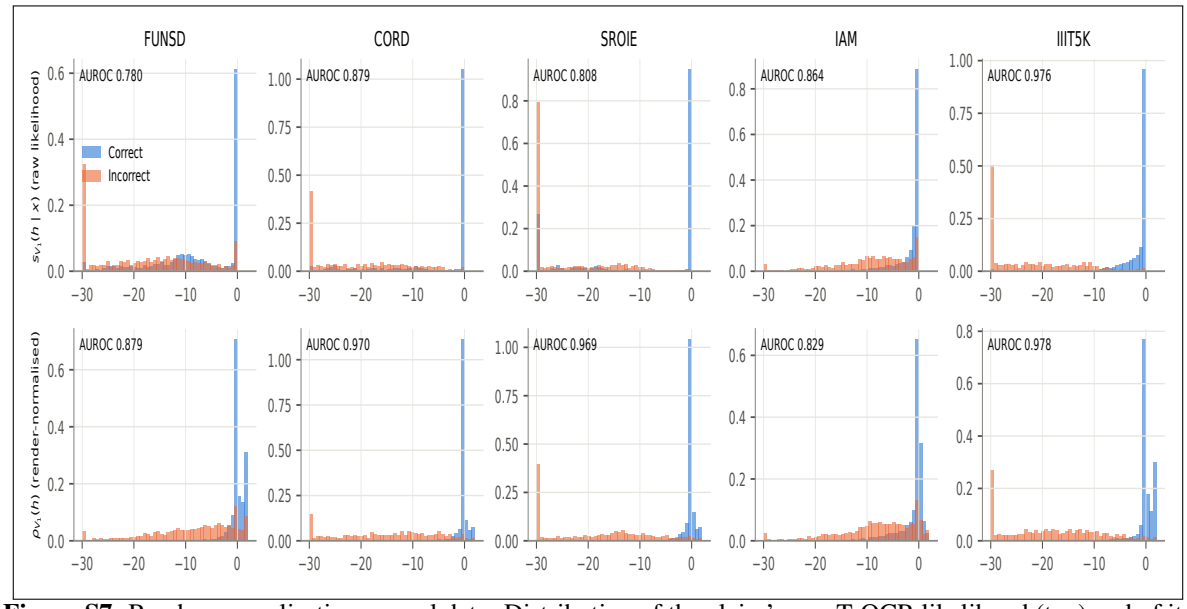

Figure S7. Render normalisation on real data. Distribution of the claim's raw TrOCR likelihood (top) and of its render-normalised ratio ρ (bottom) for correct and incorrect claims, with the AUROC of each single score.

Table S17. Single-verifier setting (no agreement features), $k = 25$, macro-F1: effect of render normalisation when only one recogniser is available.

Variant	FUNSD	CORD	SROIE	IAM	IIIT5K
TrOCR only	.930	.973	.956	.839	.957
TrOCR only, w/o render-norm.	.929	.967	.902	.849	.951
PARSeq only	.932	.968	.918	.885	.943
PARSeq only, w/o render-norm.	.933	.969	.916	.884	.947

S6. Full Results

Tables S18–S32 report every method at every k with seed standard deviations, and every metric at $k = 25$.

Table S18. FUNSD: macro-f1 as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k=5	k=10	k=25	k=50	k=100	full
Engine MSP conf.	.726	.722	.756	.764	.771	.787
ConfidNet-style	.683	.702	.771	.812	.830	.856
Flat-LR	$.897_{.041}$	$.931_{.032}$	$.948_{.009}$	$.946_{.007}$	$.945_{.011}$	$.949_{.000}$
CharEv-LR	$.870_{.033}$	$.915_{.039}$	$.935_{.005}$	$.937_{.001}$	$.940_{.002}$	$.948_{.001}$
GBDT	$.874_{.061}$	$.885_{.070}$	$.922_{.034}$	$.943_{.006}$	$.944_{.008}$	$.951_{.001}$
DeepSets	$.898_{.021}$	$.931_{.063}$	$.942_{.042}$	$.947_{.009}$	$.942_{.016}$	$.949_{.001}$
LOCUS-GNN scratch	$.868_{.033}$	$.920_{.060}$	$.933_{.031}$	$.944_{.014}$	$.938_{.022}$	$.951_{.001}$
LOCUS probe	$.921_{.013}$	$.938_{.009}$	$.946_{.014}$	$.947_{.003}$	$.947_{.007}$	$.950_{.001}$
LOCUS + CV head	$.929_{.011}$	$.934_{.008}$	$.942_{.023}$	$.946_{.004}$	$.945_{.010}$	$.949_{.001}$
Priors-only LR	$.940_{.005}$	$.946_{.002}$	$.947_{.001}$	$.947_{.001}$	$.947_{.000}$	$.946_{.000}$
LOCUS	$.923_{.020}$	$.936_{.030}$	$.943_{.007}$	$.943_{.009}$	$.946_{.004}$	$.950_{.002}$

Table S19. CORD: macro-f1 as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.742	.775	.777	.793	.799	.805
ConfidNet-style	.751	.813	.842	.882	.905	.934
Flat-LR	$.933_{.019}$	$.957_{.018}$	$.955_{.018}$	$.963_{.012}$	$.980_{.004}$	$.977_{.001}$
CharEv-LR	$.940_{.044}$	$.970_{.016}$	$.971_{.016}$	$.975_{.004}$	$.979_{.003}$	$.980_{.000}$
GBDT	$.917_{.064}$	$.960_{.016}$	$.961_{.015}$	$.970_{.008}$	$.980_{.004}$	$.980_{.001}$
DeepSets	$.949_{.033}$	$.978_{.019}$	$.977_{.015}$	$.980_{.002}$	$.980_{.003}$	$.983_{.001}$
LOCUS-GNN scratch	$.916_{.013}$	$.969_{.025}$	$.969_{.020}$	$.977_{.002}$	$.978_{.003}$	$.983_{.001}$
LOCUS probe	$.948_{.024}$	$.953_{.019}$	$.961_{.020}$	$.981_{.006}$	$.980_{.002}$	$.981_{.001}$
LOCUS + CV head	$.926_{.018}$	$.942_{.025}$	$.962_{.023}$	$.978_{.008}$	$.979_{.002}$	$.981_{.000}$
Priors-only LR	$.978_{.002}$	$.978_{.000}$	$.978_{.000}$	$.978_{.000}$	$.978_{.000}$	$.978_{.000}$
LOCUS	$.943_{.019}$	$.980_{.024}$	$.978_{.023}$	$.981_{.002}$	$.980_{.002}$	$.981_{.003}$

Table S20. SROIE: macro-f1 as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.758	.749	.780	.777	.777	.790
ConfidNet-style	.746	.785	.826	.863	.873	.912
Flat-LR	$.932_{.026}$	$.939_{.011}$	$.955_{.018}$	$.978_{.020}$	$.978_{.007}$	$.978_{.000}$
CharEv-LR	$.941_{.084}$	$.949_{.022}$	$.959_{.013}$	$.973_{.006}$	$.977_{.004}$	$.979_{.000}$
GBDT	$.956_{.019}$	$.967_{.029}$	$.962_{.035}$	$.978_{.015}$	$.979_{.003}$	$.978_{.001}$
DeepSets	$.924_{.029}$	$.930_{.025}$	$.968_{.013}$	$.969_{.007}$	$.976_{.005}$	$.978_{.000}$
LOCUS-GNN scratch	$.903_{.042}$	$.913_{.018}$	$.961_{.018}$	$.970_{.008}$	$.977_{.009}$	$.979_{.001}$
LOCUS probe	$.961_{.028}$	$.957_{.004}$	$.973_{.007}$	$.973_{.006}$	$.975_{.004}$	$.979_{.000}$
LOCUS + CV head	$.884_{.051}$	$.956_{.020}$	$.964_{.015}$	$.975_{.004}$	$.977_{.004}$	$.978_{.000}$
Priors-only LR	$.978_{.034}$	$.978_{.000}$	$.978_{.004}$	$.979_{.003}$	$.979_{.000}$	$.979_{.000}$
LOCUS	$.941_{.046}$	$.954_{.017}$	$.973_{.007}$	$.973_{.003}$	$.974_{.003}$	$.979_{.001}$

Table S21. IAM: macro-f1 as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.700	.782	.802	.801	.801	.807
ConfidNet-style	.762	.783	.801	.812	.832	.859
Flat-LR	$.863_{.038}$	$.881_{.012}$	$.904_{.017}$	$.915_{.009}$	$.918_{.003}$	$.934_{.001}$
CharEv-LR	$.838_{.020}$	$.839_{.015}$	$.887_{.016}$	$.902_{.007}$	$.920_{.008}$	$.928_{.001}$
GBDT	$.840_{.025}$	$.882_{.053}$	$.893_{.030}$	$.920_{.013}$	$.923_{.019}$	$.944_{.002}$
DeepSets	$.873_{.054}$	$.896_{.031}$	$.909_{.019}$	$.921_{.009}$	$.926_{.009}$	$.947_{.004}$
LOCUS-GNN scratch	$.865_{.039}$	$.877_{.019}$	$.908_{.023}$	$.917_{.012}$	$.927_{.008}$	$.950_{.004}$
LOCUS probe	$.870_{.037}$	$.875_{.025}$	$.904_{.016}$	$.922_{.008}$	$.931_{.007}$	$.944_{.003}$
LOCUS + CV head	$.851_{.026}$	$.888_{.030}$	$.905_{.009}$	$.916_{.009}$	$.925_{.009}$	$.944_{.003}$
Priors-only LR	$.874_{.017}$	$.897_{.013}$	$.898_{.013}$	$.908_{.003}$	$.907_{.002}$	$.907_{.001}$
LOCUS	$.869_{.020}$	$.888_{.023}$	$.906_{.011}$	$.923_{.010}$	$.927_{.006}$	$.945_{.003}$

Table S22. IIIT5K: macro-f1 as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.806	.817	.788	.817	.790	.854
ConfidNet-style	.925	.915	.922	.929	.924	.957
Flat-LR	$.966_{.021}$	$.974_{.008}$	$.969_{.005}$	$.967_{.007}$	$.970_{.009}$	$.976_{.003}$
CharEv-LR	$.958_{.022}$	$.965_{.006}$	$.965_{.008}$	$.970_{.004}$	$.967_{.008}$	$.976_{.003}$
GBDT	$.941_{.007}$	$.947_{.032}$	$.955_{.019}$	$.968_{.039}$	$.983_{.014}$	$.989_{.002}$
DeepSets	$.964_{.025}$	$.964_{.030}$	$.965_{.015}$	$.968_{.009}$	$.968_{.013}$	$.982_{.008}$
LOCUS-GNN scratch	$.961_{.020}$	$.961_{.046}$	$.962_{.022}$	$.969_{.009}$	$.973_{.009}$	$.987_{.002}$
LOCUS probe	$.966_{.010}$	$.967_{.005}$	$.968_{.008}$	$.970_{.008}$	$.966_{.011}$	$.978_{.004}$
LOCUS + CV head	$.963_{.010}$	$.966_{.008}$	$.971_{.005}$	$.969_{.011}$	$.973_{.012}$	$.978_{.003}$
Priors-only LR	$.983_{.005}$	$.982_{.004}$	$.981_{.002}$	$.980_{.003}$	$.980_{.003}$	$.977_{.003}$
LOCUS	$.974_{.003}$	$.975_{.030}$	$.971_{.017}$	$.972_{.005}$	$.973_{.010}$	$.985_{.003}$

Table S23. FUNSD: auroc as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.861	.861	.861	.861	.861	.861
ConfidNet-style	.763	.763	.845	.883	.907	.935
Flat-LR	$.965_{.013}$	$.972_{.024}$	$.975_{.003}$	$.974_{.005}$	$.975_{.006}$	$.977_{.000}$
CharEv-LR	$.943_{.034}$	$.961_{.035}$	$.973_{.003}$	$.972_{.004}$	$.973_{.003}$	$.975_{.000}$
GBDT	$.934_{.060}$	$.952_{.030}$	$.969_{.014}$	$.974_{.005}$	$.975_{.006}$	$.979_{.001}$
DeepSets	$.959_{.031}$	$.960_{.056}$	$.971_{.021}$	$.971_{.004}$	$.972_{.004}$	$.979_{.002}$
LOCUS-GNN scratch	$.954_{.020}$	$.955_{.047}$	$.969_{.010}$	$.968_{.008}$	$.970_{.005}$	$.979_{.001}$
LOCUS probe	$.956_{.014}$	$.969_{.013}$	$.975_{.003}$	$.976_{.005}$	$.976_{.006}$	$.980_{.001}$
LOCUS + CV head	$.948_{.005}$	$.966_{.005}$	$.972_{.004}$	$.975_{.007}$	$.975_{.008}$	$.979_{.001}$
Priors-only LR	$.958_{.002}$	$.967_{.002}$	$.966_{.004}$	$.966_{.004}$	$.967_{.002}$	$.968_{.001}$
LOCUS	$.964_{.010}$	$.967_{.011}$	$.974_{.004}$	$.972_{.007}$	$.975_{.003}$	$.979_{.001}$

Table S24. CORD: auroc as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.892	.892	.892	.892	.892	.892
ConfidNet-style	.827	.907	.929	.953	.964	.983
Flat-LR	$.994_{.014}$	$.995_{.002}$	$.995_{.002}$	$.995_{.000}$	$.996_{.001}$	$.997_{.000}$
CharEv-LR	$.985_{.015}$	$.992_{.004}$	$.992_{.001}$	$.993_{.001}$	$.994_{.001}$	$.996_{.000}$
GBDT	$.970_{.048}$	$.988_{.022}$	$.988_{.026}$	$.993_{.014}$	$.995_{.001}$	$.997_{.000}$
DeepSets	$.992_{.025}$	$.995_{.004}$	$.994_{.004}$	$.995_{.000}$	$.995_{.001}$	$.997_{.000}$
LOCUS-GNN scratch	$.987_{.021}$	$.992_{.019}$	$.992_{.015}$	$.995_{.001}$	$.994_{.001}$	$.997_{.001}$
LOCUS probe	$.983_{.020}$	$.991_{.013}$	$.993_{.013}$	$.994_{.001}$	$.994_{.002}$	$.996_{.001}$
LOCUS + CV head	$.982_{.004}$	$.990_{.008}$	$.993_{.008}$	$.994_{.001}$	$.995_{.000}$	$.996_{.000}$
Priors-only LR	$.994_{.000}$	$.994_{.000}$	$.994_{.000}$	$.994_{.000}$	$.994_{.000}$	$.995_{.000}$
LOCUS	$.989_{.007}$	$.993_{.005}$	$.993_{.006}$	$.994_{.001}$	$.993_{.001}$	$.996_{.001}$

Table S25. SROIE: auroc as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.859	.859	.859	.859	.859	.859
ConfidNet-style	.815	.867	.912	.941	.948	.974
Flat-LR	$.986_{.010}$	$.991_{.003}$	$.990_{.003}$	$.993_{.003}$	$.994_{.001}$	$.995_{.000}$
CharEv-LR	$.991_{.010}$	$.990_{.011}$	$.990_{.004}$	$.994_{.002}$	$.995_{.001}$	$.996_{.000}$
GBDT	$.978_{.017}$	$.986_{.028}$	$.991_{.017}$	$.994_{.003}$	$.995_{.002}$	$.997_{.000}$
DeepSets	$.983_{.019}$	$.985_{.013}$	$.992_{.004}$	$.993_{.002}$	$.993_{.001}$	$.996_{.000}$
LOCUS-GNN scratch	$.970_{.030}$	$.977_{.009}$	$.990_{.005}$	$.992_{.002}$	$.993_{.003}$	$.995_{.000}$
LOCUS probe	$.991_{.004}$	$.992_{.001}$	$.993_{.002}$	$.993_{.001}$	$.994_{.001}$	$.996_{.000}$
LOCUS + CV head	$.976_{.004}$	$.986_{.003}$	$.990_{.004}$	$.993_{.001}$	$.993_{.001}$	$.996_{.000}$
Priors-only LR	$.992_{.001}$	$.992_{.000}$	$.993_{.000}$	$.993_{.000}$	$.993_{.000}$	$.993_{.000}$
LOCUS	$.984_{.011}$	$.989_{.003}$	$.992_{.001}$	$.992_{.001}$	$.993_{.000}$	$.995_{.001}$

Table S26. IAM: auroc as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.878	.878	.878	.878	.878	.878
ConfidNet-style	.835	.849	.861	.876	.894	.917
Flat-LR	$.961_{.010}$	$.961_{.008}$	$.970_{.008}$	$.973_{.006}$	$.976_{.002}$	$.982_{.000}$
CharEv-LR	$.927_{.011}$	$.925_{.011}$	$.959_{.011}$	$.970_{.002}$	$.974_{.003}$	$.978_{.000}$
GBDT	$.929_{.027}$	$.948_{.081}$	$.964_{.012}$	$.973_{.002}$	$.976_{.006}$	$.986_{.000}$
DeepSets	$.969_{.012}$	$.960_{.021}$	$.972_{.014}$	$.975_{.010}$	$.979_{.005}$	$.986_{.001}$
LOCUS-GNN scratch	$.958_{.016}$	$.957_{.014}$	$.969_{.013}$	$.972_{.009}$	$.977_{.008}$	$.986_{.001}$
LOCUS probe	$.953_{.014}$	$.957_{.014}$	$.970_{.007}$	$.976_{.001}$	$.979_{.001}$	$.985_{.001}$
LOCUS + CV head	$.946_{.020}$	$.960_{.022}$	$.969_{.003}$	$.973_{.004}$	$.978_{.003}$	$.985_{.001}$
Priors-only LR	$.950_{.011}$	$.964_{.006}$	$.963_{.005}$	$.965_{.001}$	$.965_{.000}$	$.965_{.000}$
LOCUS	$.953_{.018}$	$.963_{.016}$	$.969_{.007}$	$.974_{.005}$	$.978_{.004}$	$.986_{.000}$

Table S27. IIIT5K: auroc as a function of k (labelled words per class). Ensemble of five seeds; subscript: std. of single-seed runs.

Method	k**=5**	k**=10**	k**=25**	k**=50**	k**=100**	**full**
Engine MSP conf.	.938	.938	.938	.938	.938	.938
ConfidNet-style	.982	.981	.980	.984	.985	.988
Flat-LR	$.990_{.002}$	$.991_{.001}$	$.991_{.002}$	$.992_{.002}$	$.993_{.001}$	$.994_{.000}$
CharEv-LR	$.990_{.002}$	$.991_{.000}$	$.993_{.006}$	$.994_{.002}$	$.994_{.002}$	$.997_{.000}$
GBDT	$.976_{.012}$	$.982_{.012}$	$.990_{.026}$	$.994_{.003}$	$.996_{.000}$	$.995_{.000}$
DeepSets	$.990_{.003}$	$.988_{.007}$	$.990_{.005}$	$.994_{.005}$	$.995_{.003}$	$.995_{.001}$
LOCUS-GNN scratch	$.987_{.004}$	$.987_{.007}$	$.990_{.002}$	$.994_{.001}$	$.995_{.001}$	$.995_{.001}$
LOCUS probe	$.991_{.002}$	$.992_{.001}$	$.993_{.001}$	$.993_{.001}$	$.995_{.002}$	$.996_{.000}$
LOCUS + CV head	$.984_{.002}$	$.985_{.006}$	$.994_{.002}$	$.994_{.002}$	$.995_{.001}$	$.995_{.001}$
Priors-only LR	$.991_{.000}$	$.991_{.000}$	$.991_{.000}$	$.991_{.000}$	$.991_{.000}$	$.991_{.001}$
LOCUS	$.986_{.002}$	$.989_{.002}$	$.991_{.002}$	$.992_{.001}$	$.994_{.002}$	$.995_{.001}$

Table S28. All metrics on FUNSD at $k = 25$ (ensemble of five seeds for learned methods; mean over seeds for threshold baselines). †oracle threshold tuned on test.

Method	Macro-F1	$F1_{err}$	Acc.	AUROC	$AUPRC_{err}$	AURC↓	Cov@95
Engine MSP confidence	.756	.734	.759	.861	.750	.144	.346
ConfidNet-style learned conf.[7]	.771	.723	.781	.845	.805	.164	.053
Two-engine voting	.609	.641	.612	.680	.479	.220	.021
Re-OCR: TrOCR-base[4]	.605	.636	.608	.675	.475	.229	.000
Re-OCR: Florence-2[5]	.647	.670	.649	.715	.508	.198	.311
Re-OCR: PARSeq[3]	.946	.931	.951	.939	.907	.125	.006
Re-OCR: PARSeq, graded	.946	.931	.951	.976	.970	.090	.646
VLM judge: Qwen2.5-VL-7B[6]	.760	.683	.785	.846	.764	.153	.147
VLM judge, oracle threshold†	.762	.684	.789	.846	.764	.153	.147
LR on flattened graph evidence	.948	.933	.952	.975	.973	.092	.645
LR on character evidence	.935	.916	.940	.973	.969	.094	.631
GBDT on all evidence	.922	.903	.927	.969	.964	.098	.625
DeepSets over rivals (no edges)	.942	.926	.947	.971	.968	.094	.639
LOCUS-GNN from scratch	.933	.914	.938	.969	.966	.095	.630
LOCUS zero-shot (no labels)	.918	.890	.927	.949	.953	.110	.270
LOCUS (k-shot)	.943	.926	.948	.974	.971	.092	.628
LOCUS + CV-selected head	.942	.926	.947	.972	.971	.094	.642
LOCUS linear probe	.946	.930	.950	.975	.973	.093	.639
Fusion of label-free priors (LR)	.947	.932	.952	.966	.968	.101	.643
LOCUS rule (zero-shot, ρ margin)	.866	.836	.873	.943	.939	.110	.471

Table S29. All metrics on CORD at $k = 25$ (ensemble of five seeds for learned methods; mean over seeds for threshold baselines). †oracle threshold tuned on test.

Method	Macro-F1	$F1_{err}$	Acc.	AUROC	$AUPRC_{err}$	AURC↓	Cov@95
Engine MSP confidence	.777	.759	.781	.892	.839	.140	.408
ConfidNet-style learned conf.[7]	.842	.818	.846	.929	.897	.124	.465
Two-engine voting	.772	.775	.772	.807	.635	.170	.409
Re-OCR: TrOCR-base[4]	.811	.807	.812	.841	.679	.154	.447

Re-OCR: Florence-2[5]	.615	.677	.625	.685	.514	.233	.249
Re-OCR: PARSeq[3]	.973	.968	.974	.975	.945	.109	.615
Re-OCR: PARSeq, graded	.973	.968	.974	.986	.970	.099	.614
VLM judge: Qwen2.5-VL-7B[6]	.891	.862	.899	.945	.943	.127	.351
VLM judge, oracle threshold†	.902	.879	.908	.945	.943	.127	.351
LR on flattened graph evidence	.955	.945	.958	.995	.995	.096	.625
LR on character evidence	.971	.965	.972	.992	.991	.098	.624
GBDT on all evidence	.961	.953	.963	.988	.981	.100	.621
DeepSets over rivals (no edges)	.977	.972	.978	.994	.994	.097	.627
LOCUS-GNN from scratch	.969	.963	.970	.992	.991	.098	.624
LOCUS zero-shot (no labels)	.904	.887	.907	.977	.972	.104	.551
LOCUS (k-shot)	.978	.974	.979	.993	.992	.097	.626
LOCUS + CV-selected head	.962	.953	.963	.993	.993	.097	.623
LOCUS linear probe	.961	.952	.963	.993	.992	.099	.625
Fusion of label-free priors (LR)	.978	.974	.979	.994	.994	.097	.626
LOCUS rule (zero-shot, ρ margin)	.934	.923	.935	.991	.990	.098	.618

Table S30. All metrics on SROIE at $k = 25$ (ensemble of five seeds for learned methods; mean over seeds for threshold baselines). †oracle threshold tuned on test.

Method	**Macro-F1**	**F1**$_{err}$	**Acc.**	**AUROC**	**AUPRC**$_{err}$	**AURC**↓	**Cov@95**
Engine MSP confidence	.780	.766	.781	.859	.826	.211	.091
ConfidNet-style learned conf.[7]	.826	.822	.826	.912	.903	.184	.233
Two-engine voting	.976	.975	.976	.976	.963	.152	.547
Re-OCR: TrOCR-base[4]	.979	.977	.979	.978	.969	.151	.550
Re-OCR: Florence-2[5]	.478	.676	.553	.578	.512	.377	.092
Re-OCR: PARSeq[3]	.742	.787	.750	.763	.653	.246	.305
Re-OCR: PARSeq, graded	.742	.787	.750	.760	.681	.242	.304
VLM judge: Qwen2.5-VL-7B[6]	.866	.845	.870	.941	.948	.172	.238
VLM judge, oracle threshold†	.883	.870	.885	.941	.948	.172	.238

LR on flattened graph evidence	.955	.951	.956	.990	.991	.139	.546
LR on character evidence	.959	.956	.960	.990	.991	.139	.545
GBDT on all evidence	.962	.959	.962	.991	.991	.138	.548
DeepSets over rivals (no edges)	.968	.965	.968	.992	.993	.138	.552
LOCUS-GNN from scratch	.961	.958	.962	.990	.992	.139	.550
LOCUS zero-shot (no labels)	.798	.823	.801	.966	.968	.152	.401
LOCUS (k-shot)	.973	.971	.973	.992	.993	.139	.553
LOCUS + CV-selected head	.964	.961	.964	.990	.991	.139	.548
LOCUS linear probe	.973	.971	.973	.993	.994	.138	.555
Fusion of label-free priors (LR)	.978	.977	.979	.993	.994	.138	.553
LOCUS rule (zero-shot, ρ margin)	.910	.910	.911	.981	.981	.144	.507

Table S31. All metrics on IAM at $k = 25$ (ensemble of five seeds for learned methods; mean over seeds for threshold baselines). †oracle threshold tuned on test.

Method	**Macro-F1**	**F1**$_{err}$	**Acc.**	**AUROC**	**AUPRC**$_{err}$	**AURC**$\downarrow$	**Cov@95**
Engine MSP confidence	.802	.791	.804	.878	.859	.203	.091
ConfidNet-style learned conf.[7]	.801	.784	.803	.861	.865	.226	.007
Two-engine voting	.759	.769	.760	.767	.663	.272	.001
Re-OCR: TrOCR-base[4]	.758	.769	.759	.766	.661	.273	.001
Re-OCR: Florence-2[5]	.369	.643	.488	.522	.475	.424	.001
Re-OCR: PARSeq[3]	.879	.881	.879	.885	.799	.186	.455
Re-OCR: PARSeq, graded	.879	.881	.879	.914	.877	.168	.454
VLM judge: Qwen2.5-VL-7B[6]	.700	.713	.700	.788	.765	.250	.052
VLM judge, oracle threshold†	.719	.694	.722	.788	.765	.250	.052
LR on flattened graph evidence	.904	.895	.905	.970	.968	.146	.483
LR on character evidence	.887	.878	.888	.959	.956	.152	.446
GBDT on all evidence	.893	.884	.894	.964	.961	.150	.467
DeepSets over rivals (no edges)	.909	.901	.910	.972	.971	.147	.497
LOCUS-GNN from scratch	.908	.899	.909	.969	.970	.149	.482

LOCUS zero-shot (no labels)	.825	.814	.826	.909	.895	.180	.346
LOCUS (k-shot)	.906	.896	.907	.969	.968	.149	.486
LOCUS + CV-selected head	.905	.898	.906	.969	.967	.146	.490
LOCUS linear probe	.904	.895	.905	.970	.970	.148	.473
Fusion of label-free priors (LR)	.898	.891	.899	.963	.959	.149	.481
LOCUS rule (zero-shot, ρ margin)	.844	.838	.844	.933	.934	.167	.330

Table S32. All metrics on IIIT5K at $k = 25$ (ensemble of five seeds for learned methods; mean over seeds for threshold baselines). †oracle threshold tuned on test.

Method	**Macro-F1**	**F1**$_{err}$	**Acc.**	**AUROC**	**AUPRC**$_{err}$	**AURC**$\downarrow$	**Cov@95**
Engine MSP confidence	.788	.738	.801	.938	.857	.069	.631
ConfidNet-style learned conf.[7]	.922	.892	.933	.980	.958	.054	.726
Two-engine voting	.862	.819	.876	.905	.696	.076	.635
Re-OCR: TrOCR-base[4]	.952	.934	.960	.966	.879	.058	.715
Re-OCR: Florence-2[5]	.823	.776	.836	.879	.635	.083	.595
Re-OCR: PARSeq[3]	.993	.990	.994	.991	.984	.050	.746
Re-OCR: PARSeq, graded	.993	.990	.994	.995	.994	.049	.747
VLM judge: Qwen2.5-VL-7B[6]	.888	.850	.901	.974	.943	.056	.716
VLM judge, oracle threshold†	.928	.899	.940	.974	.943	.056	.716
LR on flattened graph evidence	.969	.956	.974	.991	.988	.052	.745
LR on character evidence	.965	.951	.971	.993	.990	.050	.745
GBDT on all evidence	.955	.937	.962	.990	.987	.052	.746
DeepSets over rivals (no edges)	.965	.951	.971	.990	.987	.052	.747
LOCUS-GNN from scratch	.962	.946	.968	.990	.987	.052	.747
LOCUS zero-shot (no labels)	.968	.955	.973	.983	.954	.054	.739
LOCUS (k-shot)	.971	.959	.976	.991	.985	.052	.744
LOCUS + CV-selected head	.971	.960	.976	.994	.992	.049	.747
LOCUS linear probe	.968	.956	.974	.993	.991	.051	.748
Fusion of label-free priors (LR)	.981	.974	.985	.991	.985	.052	.747

LOCUS rule (zero-shot, ρ margin)	.790	.742	.801	.981	.963	.055	.735

S7. Extended Results Referenced in the Main Article

The following tables and figures belong to the main article and were moved here to respect its page limit; the main text discusses each of them.

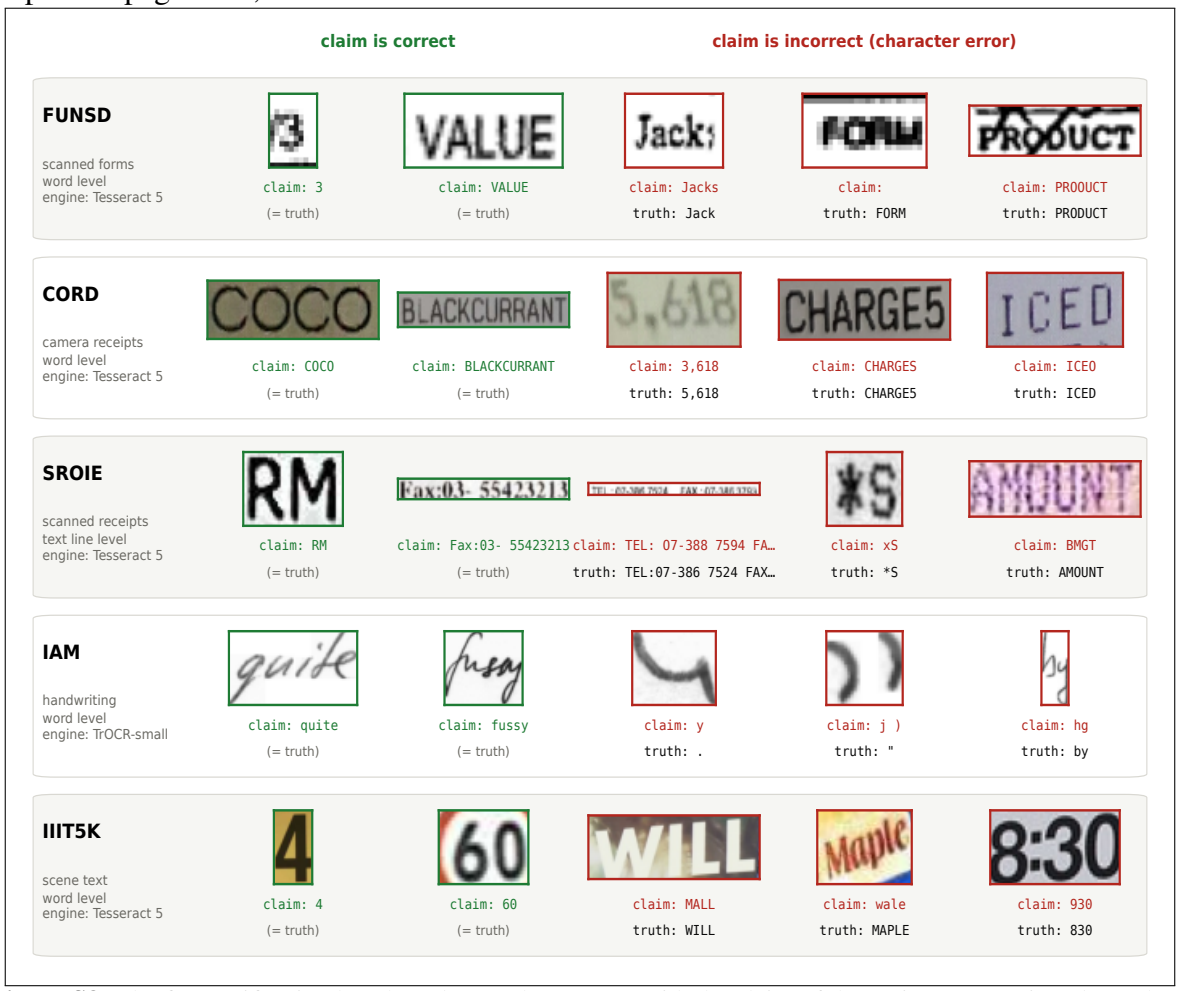

Figure S8. The five verification benchmarks. Real test crops with the claim of the main OCR engine, drawn by a seeded random choice (two correct claims and three character errors per dataset; none chosen by hand). Errors range from single confusable glyphs (`CHARGES`/`CHARGE5`, `3,618`/`5,618`) to empty or noise outputs.

Table S33. Counterfactual graphs on the test sets: average hypotheses per claim and share of real engine errors whose ground truth equals TrOCR's reading, PARSeq's reading, or *any* node of $G(h)$.

Dataset	**nodes**	**truth** $= g_1$ **(%)**	**truth** $= g_2$ **(%)**	**truth** $\in G(h)$ **(%)**
FUNSD	17.2	31.3	72.4	82.8
CORD	15.3	70.4	89.5	94.9
SROIE	21.2	89.5	44.1	92.6
IAM	17.2	14.4	46.5	55.4
IIIT5K	15.9	86.8	95.3	97.0

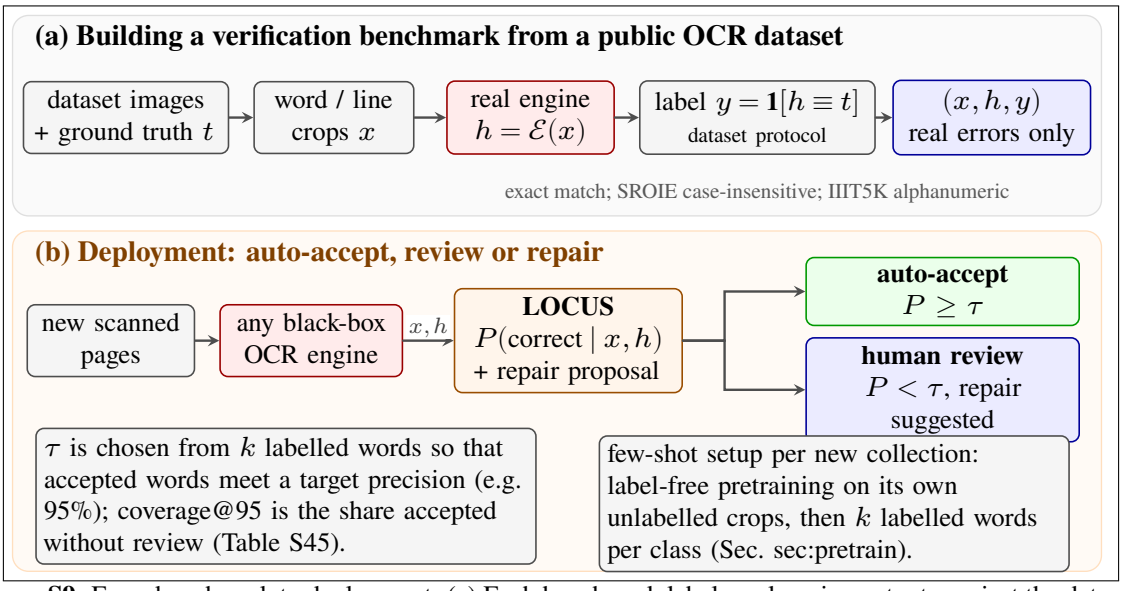

Figure S9. From benchmark to deployment. (a) Each benchmark labels real engine outputs against the dataset's own ground truth under its protocol. (b) In a digitisation pipeline, words above a precision-targeted threshold are accepted automatically and the rest go to a reviewer with a repair suggestion.

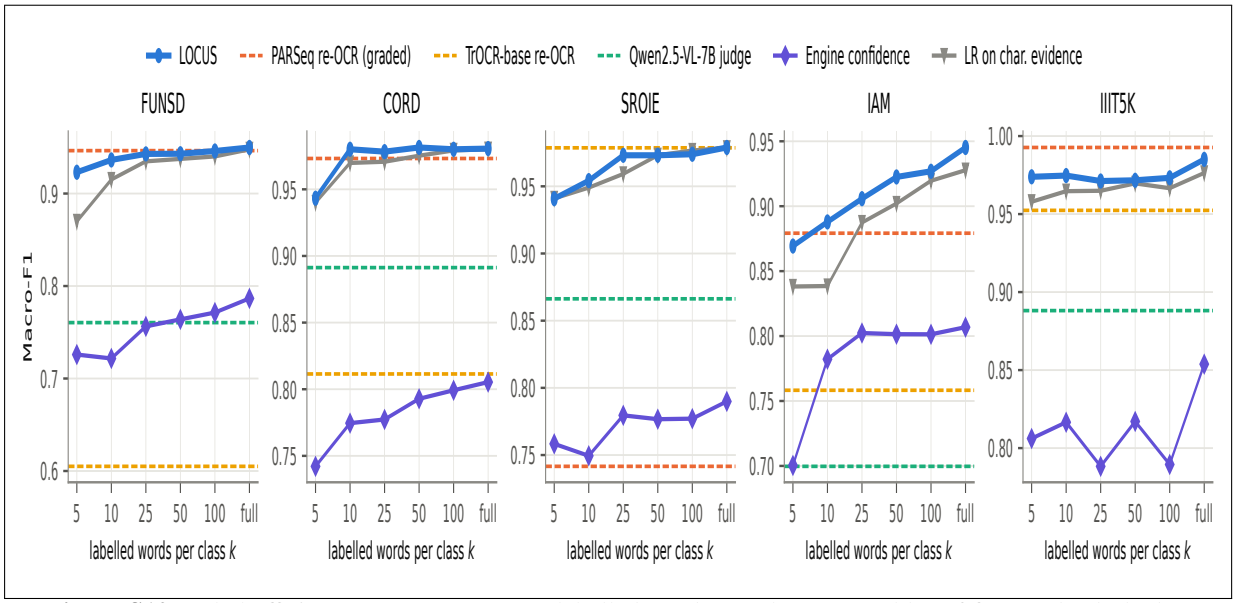

Figure S10. Label efficiency. Macro-F1 versus labelled words per class (ensembles of five seeds; dashed: label-free baselines). AUROC curves: Figure S2.

Table S34. Render normalisation by claim type: AUROC of TrOCR's raw claim likelihood $s(h|x)$ (raw) and of the render-normalised ratio $\rho(h)$ (ρ), on test claims with digits, upper-case words and lower-/mixed-case words (subsets with fewer than 50 claims omitted).

Dataset	digits		upper-case		lower/mixed	
	raw	ρ	raw	ρ	raw	ρ
FUNSD	0.932	0.911	0.950	0.951	0.714	0.807
CORD	0.981	0.981	0.984	0.988	0.752	0.948
SROIE	0.901	0.976	1.000	1.000	0.722	0.972
IAM	0.740	0.638	–	–	0.849	0.814
IIIT5K	0.940	0.903	0.986	0.985	0.964	0.977

Table S35. Transfer. Top: engine transfer (train on one engine's claims, test on the other's; in-engine: trained and tested on the target engine). Right: dataset transfer, macro-F1 (Bottom: training dataset, fully supervised; shaded: in-domain).

Dataset	Train→Test	$k = 25$			full		
		F1 in	F1 tr.	AUROC tr.	F1 in	F1 tr.	AUROC tr.
FUNSD	Tesseract→TrOCR-s	.955	.838	.963	.959	.958	.988
FUNSD	TrOCR-s→Tesseract	.941	.927	.964	.951	.947	.976
CORD	Tesseract→TrOCR-s	.956	.689	.974	.971	.891	.989
CORD	TrOCR-s→Tesseract	.980	.817	.982	.979	.879	.993
SROIE	Tesseract→TrOCR-s	.629	.745	.862	.716	.773	.884
SROIE	TrOCR-s→Tesseract	.973	.776	.922	.979	.787	.941
IIIT5K	Tesseract→TrOCR-s	.990	.980	.995	.984	.993	.996
IIIT5K	TrOCR-s→Tesseract	.971	.972	.988	.985	.966	.991

Train\Test	FUNSD	CORD	SROIE	IAM	IIIT5K
FUNSD	.949	.977	.861	.905	.938
CORD	.942	.981	.959	.882	.914
SROIE	.620	.823	.978	.779	.946
IAM	.880	.878	.734	.944	.925
IIIT5K	.946	.973	.767	.899	.978

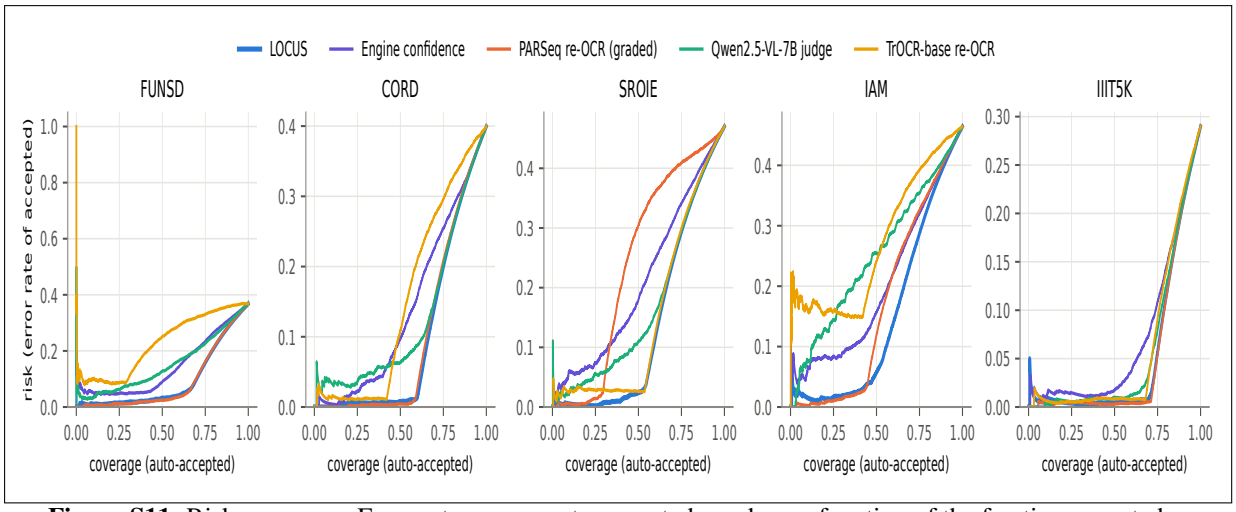

Figure S11. Risk–coverage. Error rate among auto-accepted words as a function of the fraction accepted ($k = 25$; lower is better).

Table S36. Recogniser dependence. Word accuracy (%) of each verifier's reading, the better re-recognition baseline, and LOCUS ($k = 25$) against it.

Dataset	verifier accuracy		best	macro-F1			AUROC		
	TrOCR	PARSeq	baseline	base	LOCUS	Δ	base	LOCUS	Δ
FUNSD	38.0	88.9	PARSeq	0.946	0.943	-0.004	0.976	0.974	-0.002
CORD	69.8	94.0	PARSeq	0.973	0.978	+0.005	0.986	0.993	+0.007
SROIE	94.3	49.5	TrOCR-base	0.979	0.973	-0.006	0.978	0.992	+0.014
IAM	42.4	64.7	PARSeq	0.879	0.906	+0.027	0.914	0.969	+0.055
IIIT5K	92.7	98.5	PARSeq	0.993	0.971	-0.021	0.995	0.991	-0.004

Table S37. Word-level intervals inflated by the worst-case design effect (Deff) estimated on FUNSD and CORD. sig.: 95% interval excludes zero.

Dataset	Deff	Baseline	ΔAUROC	95% CI	sig.	ΔF1	95% CI	sig.
SROIE	3.67	PARSeq re-OCR (graded)	+0.232	[+0.204, +0.260]	yes	+0.232	[+0.204, +0.259]	yes
SROIE	3.67	TrOCR-base re-OCR	+0.014	[+0.007, +0.020]	yes	-0.005	[-0.013, +0.002]	no
IAM	2.29	PARSeq re-OCR (graded)	+0.055	[+0.042, +0.067]	yes	+0.026	[+0.009, +0.044]	yes
IAM	2.29	TrOCR-base re-OCR	+0.203	[+0.185, +0.221]	yes	+0.148	[+0.127, +0.168]	yes

Table S38. Rendering sensitivity: AUROC of the claim's ratio $\rho_{V_1} + \rho_{V_2}$ alone, AUROC of the training-free ρ-margin rule, and macro-F1 of LOCUS trained on 1,000 labelled words (3 seeds), for each rendering.

Rendering	FUNSD			CORD			IAM		
	$\rho(h)$	rule	LOCUS	$\rho(h)$	rule	LOCUS	$\rho(h)$	rule	LOCUS
Liberation Sans (default)	.903	.923	.937	.981	.996	.972	.921	.921	.921
Liberation Serif	.906	.928	.938	.984	.995	.972	.918	.944	.920
Liberation Mono	.907	.940	.938	.990	.994	.977	.915	.931	.923
DejaVu Sans	.912	.920	.937	.984	.995	.976	.922	.923	.919
Comic Neue (handwriting-like)	.907	.928	.939	.989	.993	.976	.917	.937	.921
Liberation Sans, glyphs 0.6x	.903	.924	.939	.983	.996	.976	.915	.919	.923
Liberation Sans, glyphs 0.9x	.901	.924	.937	.980	.995	.973	.923	.923	.921

Table S39. AUROC on all errors / on genuine character errors only ($k = 25$). Removed: share of incorrect claims that differ from the ground truth only in case or punctuation.

Dataset	removed	LOCUS	PARSeq graded	TrOCR re-OCR	Qwen2.5-VL-7B	Engine conf.
FUNSD	51%	0.974 / 0.976	0.976 / 0.979	0.675 / 0.691	0.846 / 0.928	0.861 / 0.886
CORD	19%	0.993 / 0.996	0.986 / 0.987	0.841 / 0.845	0.945 / 0.989	0.892 / 0.907
SROIE	23%	0.992 / 0.999	0.760 / 0.756	0.978 / 0.991	0.941 / 0.982	0.859 / 0.886
IAM	18%	0.969 / 0.980	0.914 / 0.908	0.766 / 0.802	0.788 / 0.818	0.878 / 0.916
IIIT5K	0%	0.991 / 0.991	0.995 / 0.995	0.966 / 0.966	0.974 / 0.974	0.938 / 0.938

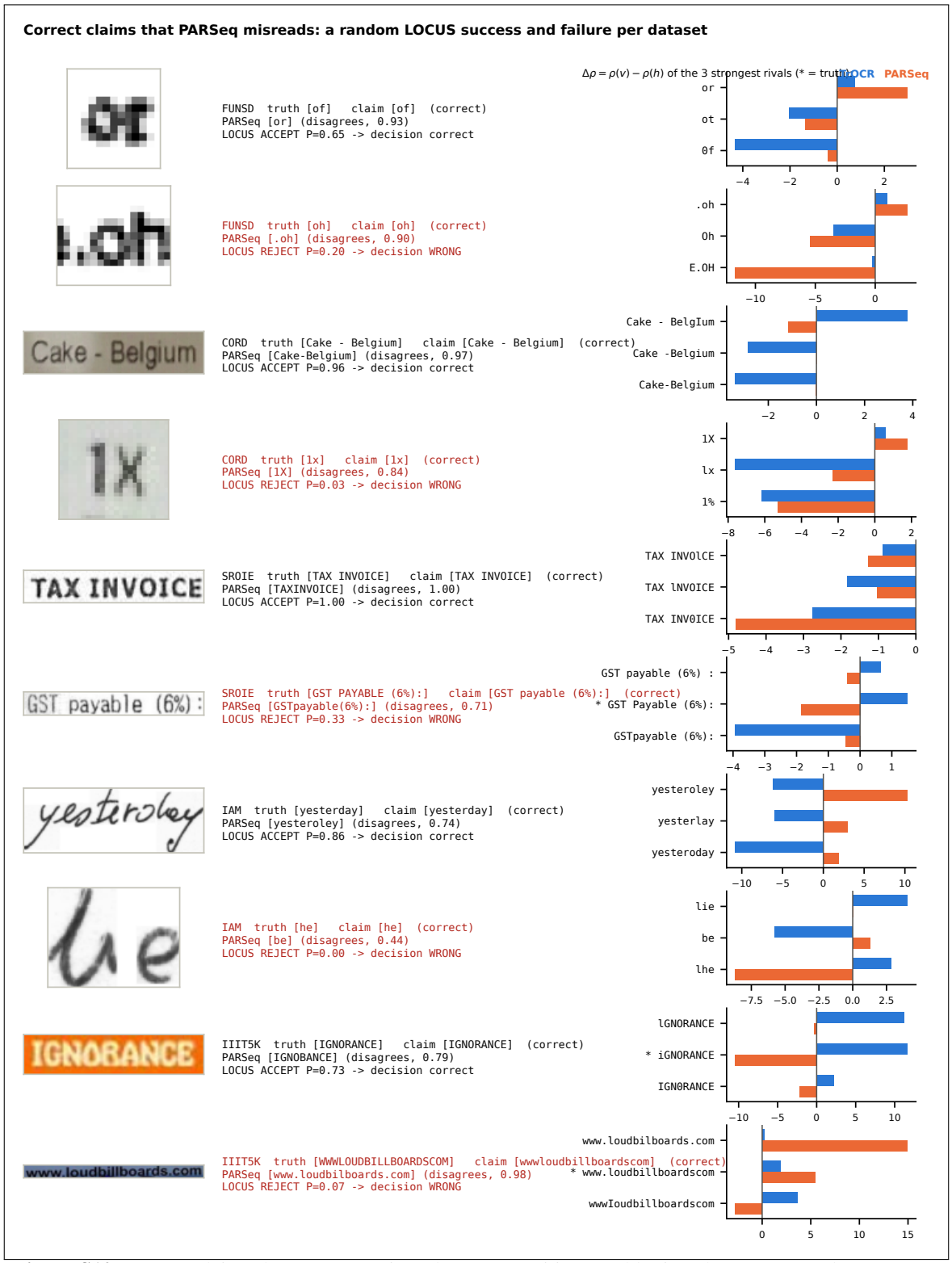

Figure S12. Correct claims that PARSeq misreads (re-recognition would reject them): one random LOCUS success and one random failure per dataset. Comparing the claim with its rivals under two verifiers helps most in this regime: a rival that one verifier prefers is typically rejected by the other.

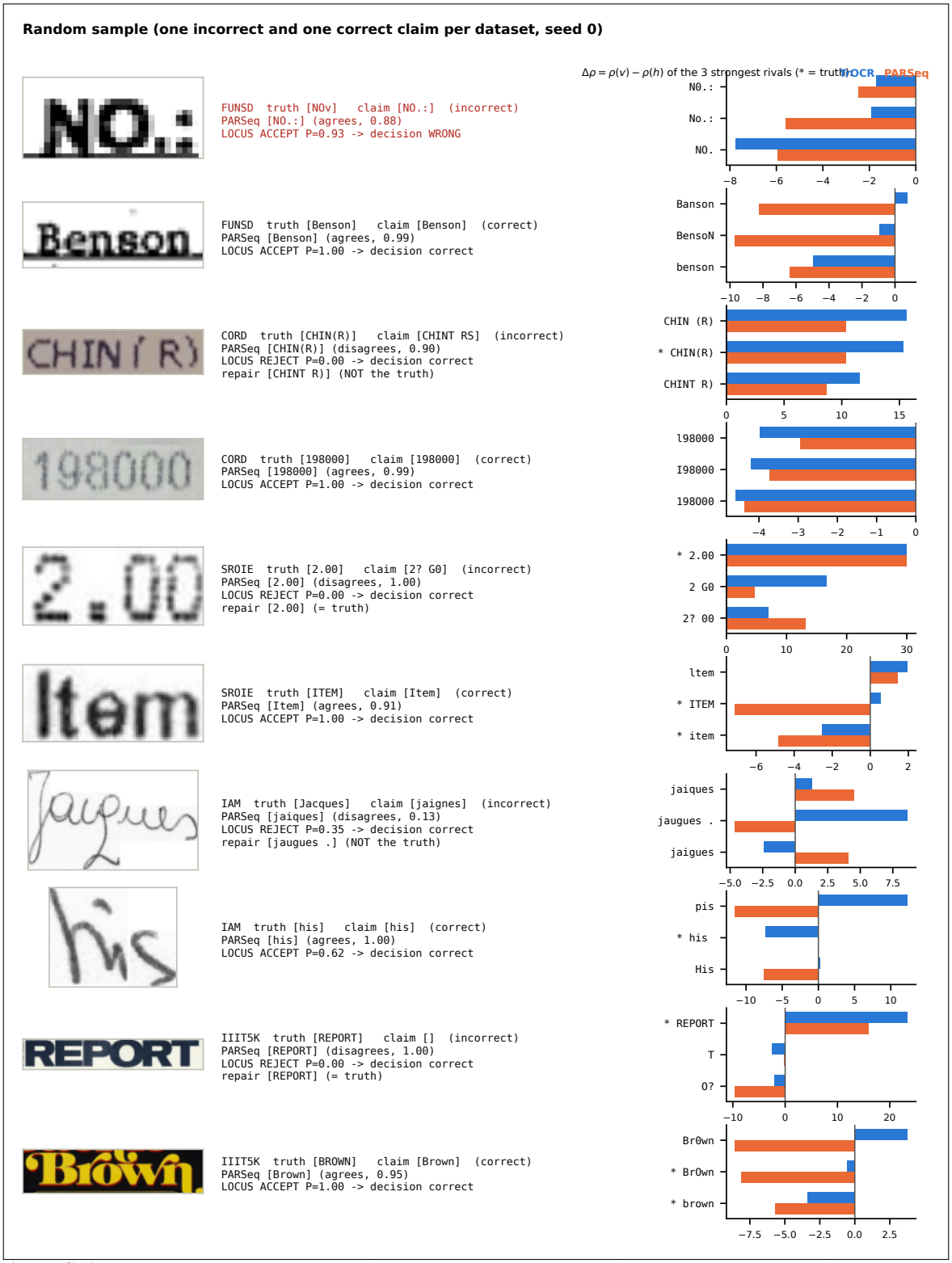

Figure S13. Random sample (seed 0): one incorrect and one correct claim per dataset, drawn without regard to LOCUS' output; wrong decisions in red. Annotation errors also appear (e.g. FUNSD ground truth "NOv" for a crop reading "NO.:"; Table S41).

Table S40. Regimes on the full test sets ($k = 25$). Artefacts: incorrect claims that differ from the ground truth only in case or punctuation. Right: errors PARSeq agrees with, restricted to genuine character errors.

Dataset	all errors		correct claims PARSeq misreads		character errors PARSeq agrees with	
	count	artefacts	count	LOCUS accepts	count	LOCUS rejects
FUNSD	1474	751 (51%)	38	14 (37%)	47	3 (6%)
CORD	941	181 (19%)	43	28 (65%)	9	1 (11%)
SROIE	1880	438 (23%)	971	953 (98%)	7	4 (57%)
IAM	1857	326 (18%)	419	312 (74%)	53	14 (26%)
IIIT5K	869	0 (0%)	4	3 (75%)	14	0 (0%)

Table S41. Manual audit of random test words (20 per cell, fewer where the stratum is smaller). G: genuine, N: label noise, C: crop problem, A: ambiguous, P: claim correct only under the dataset's normalisation protocol although its exact appearance differs from the image (IIIT5K only).

Dataset	LOCUS decision wrong					errors PARSeq agrees with			
	G	N	C	A	P	G	N	C	A
FUNSD	8	6	4	2	0	0	16	3	1
CORD	10	3	4	3	0	1	8	4	5
SROIE	7	12	1	0	0	5	10	1	4
IAM	16	1	0	3	0	4	1	11	4
IIIT5K	1	3	0	0	16	0	11	2	1

Table S42. Decision accuracy (%) stratified by which verifier reads the word correctly ($k = 25$; err.: share of incorrect claims in the stratum; strata with fewer than 20 words omitted).

Dataset	stratum	words	err. (%)	LOCUS	PARSeq agree.	TrOCR agree.
FUNSD	both right	1475	29	99.5	100.0	100.0
	only TrOCR right	44	80	70.5	47.7	100.0
	only PARSeq right	2080	31	98.7	100.0	29.0
	neither right	401	93	60.1	56.9	76.6
CORD	both right	1592	39	100.0	100.0	100.0
	only TrOCR right	52	71	92.3	61.5	100.0
	only PARSeq right	622	35	97.9	100.0	34.2
	neither right	90	69	64.4	54.4	61.1
SROIE	both right	1946	41	99.5	100.0	100.0
	only TrOCR right	1827	48	98.1	47.0	100.0
	only PARSeq right	33	88	63.6	100.0	51.5
	neither right	194	87	73.2	82.5	64.4
IAM	both right	1400	13	99.3	100.0	100.0
	only TrOCR right	296	27	89.5	25.7	100.0
	only PARSeq right	1187	57	89.0	100.0	43.6
	neither right	1117	82	81.9	76.5	73.6
IIIT5K	both right	2764	27	98.9	100.0	100.0
	only TrOCR right	15	–	–	–	
	only PARSeq right	190	45	86.3	100.0	42.6
	neither right	30	97	50.0	53.3	60.0

Table S43. Message passing versus counterfactual evidence ($k = 25$, macro-F1 / AUROC, ensembles of five seeds; best in bold, second best underlined). Claim-relative features: the differences $\phi(v) - \phi(h)$ between each rival and the claim and the claim's agreement with the verifiers' readings.

Variant	FUNSD	CORD	SROIE	IAM	IIIT5K
A full	.943 / .974	.979 / .993	.973 / .992	.901 / .967	.970 / .991
B no message passing	.945 / .975	.978 / .994	.970 / .992	.895 / .966	.979 / .991
C no claim-relative features	.936 / .972	.969 / .993	.950 / .988	.898 / .964	.954 / .988
D no claim-relative features, no MP	.943 / .974	.967 / .994	.956 / .990	.882 / .960	.960 / .987
E claim node only (no rivals)	.944 / .961	.978 / .993	.972 / .992	.914 / .967	.962 / .990

Table S44. Words misread by both verifiers ($k = 25$). err.: share of incorrect claims (%); accuracy (%) of LOCUS, PARSeq agreement and TrOCR agreement; AUROC of LOCUS (cells with fewer than 10 words omitted).

Dataset	stratum	words	err.	LOCUS	PARSeq	TrOCR	AUROC
FUNSD	both wrong, same reading	144	93.8	56.2	51.4	51.4	0.500
	of which claim = that reading	61	100.0	11.5	0.0	0.0	–
	both wrong, different readings	257	92.2	62.3	59.9	90.7	0.638
CORD	both wrong, same reading	47	72.3	57.4	57.4	57.4	0.681
	of which claim = that reading	7	–	–	–	–	–
	both wrong, different readings	43	65.1	72.1	51.2	65.1	0.726
SROIE	both wrong, same reading	27	70.4	51.9	40.7	40.7	0.579
	of which claim = that reading	8	–	–	–	–	–
	both wrong, different readings	167	89.2	76.6	89.2	68.3	0.770
IAM	both wrong, same reading	70	80.0	48.6	44.3	44.3	0.513
	of which claim = that reading	25	100.0	8.0	0.0	0.0	–
	both wrong, different readings	1047	81.9	84.1	78.6	75.5	0.877
IIIT5K	both wrong, same reading	18	100.0	38.9	38.9	38.9	–
	of which claim = that reading	11	100.0	0.0	0.0	0.0	–
	both wrong, different readings	12	91.7	66.7	75.0	91.7	0.727

Table S45. Human-review workload: percentage of all words that must be reviewed so that the automatically accepted words reach the target precision ($k = 25$; lower is better).

Dataset	Method	90%	95%	98%
FUNSD	LOCUS	30.1	37.2	61.1
	PARSeq re-OCR (graded)	30.4	35.4	56.2
	TrOCR-base re-OCR	69.8	100.0	100.0
	Qwen2.5-VL-7B judge	57.1	85.3	100.0
	Engine confidence	47.9	65.5	99.9
CORD	LOCUS	33.6	37.4	39.6
	PARSeq re-OCR (graded)	34.8	38.6	40.6
	TrOCR-base re-OCR	50.8	55.3	57.3
	Qwen2.5-VL-7B judge	36.7	64.9	95.5
	Engine confidence	49.4	59.2	76.0
SROIE	LOCUS	41.2	44.7	54.9
	PARSeq re-OCR (graded)	66.9	69.5	71.7
	TrOCR-base re-OCR	41.9	45.0	94.6
	Qwen2.5-VL-7B judge	53.0	76.2	92.3
	Engine confidence	67.9	90.9	97.0

Dataset	Method	90%	95%	98%
IAM	LOCUS	44.3	51.4	66.9
	PARSeq re-OCR (graded)	51.3	54.6	62.5
	TrOCR-base re-OCR	99.9	99.9	99.9
	Qwen2.5-VL-7B judge	86.9	94.8	96.8
	Engine confidence	63.9	90.9	99.8
IIIT5K	LOCUS	21.3	25.6	28.1
	PARSeq re-OCR (graded)	21.1	25.3	27.7
	TrOCR-base re-OCR	24.2	28.5	31.0
	Qwen2.5-VL-7B judge	23.0	28.4	34.2
	Engine confidence	26.1	36.9	47.8

S8. Reproduction Guide

All steps are scripts in the accompanying code; every table and figure is regenerated from raw result files.

1. **Environment:** Python 3.10, PyTorch 2 with CUDA, PyTorch Geometric, Hugging Face `transformers` (TrOCR, Florence-2, Qwen2.5-VL), PARSeq from its official release, Tesseract 5 and the Liberation fonts.
2. **Data:** Obtain FUNSD, CORD, SROIE, IAM and IIIT5K from their owners; `download.py` and `extract_crops.py` rebuild the crops listed in the released manifests (identical crop ids, splits and seeds).
3. **Claims:** `ocr_hyp.py` runs Tesseract and TrOCR-small and writes the claims and confidences.
4. **Evidence:** `locus2_features.py` builds the counterfactual graphs and the render-normalised evidence under TrOCR-base and PARSeq (one GPU, about 18 words/s); `features.py` computes the character evidence.
5. **Baselines:** `reocr_baselines.py`, `vlm_judge.py` and `eval_external.py`.
6. **LOCUS:** `locus.py main` (pretraining, k-shot adaptation, ensembles), `locus.py ablate`, `xengine` and `xdataset`; the revision analyses are `revision_exp.py` (noise, graph size, pseudo-labels, recalibration), `revision_beam.py`, revision_bench.py and `revision_analysis.py`.
7. **Tables and Figures:** `make_assets.py`, `make_analysis.py`, revision_tables.py, then the LaTeX build script. Each result row in `results/*.jsonl` records dataset, method, k, seed and all metrics.