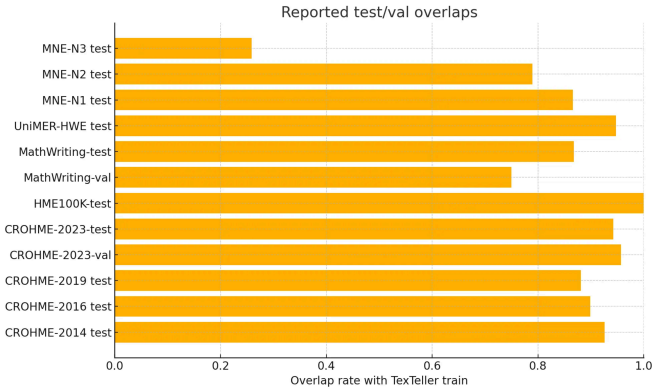
Comment on arxiv 2508.09220: Data Leakage Audit of TexTeller on Public Benchmarks

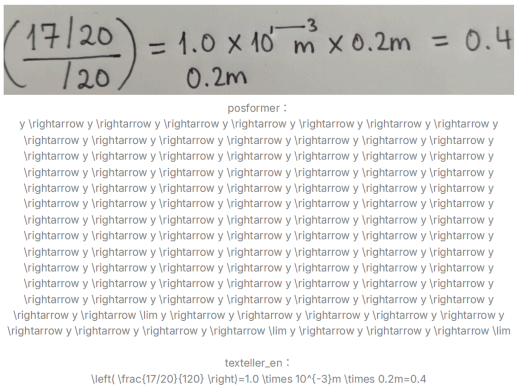
Haoran Wang[1]

TexTeller, a recently proposed method for Handwritten Mathematical Expression Recognition (HMER), has reported unprecedented accuracy on several public benchmarks. This audit reveals that these state-of-the-art (SOTA) claims are predicated on severe and systemic data leakage, where the model's training data was contaminated with test sets from these same benchmarks. We provide a three-pronged evidentiary analysis. First, we present a quantitative audit demonstrating near-total train-test overlap; for instance, 100.0% of the HME100K test set and 94.2% of the CROHME 2023 test set were found verbatim in TexTeller's training corpus. Second, we identify consequential performance anomalies inconsistent with valid generalization, such as test accuracy paradoxically exceeding training accuracy. Third, we document direct evidence of memorization through "error-label memory," where the model precisely reproduces annotation errors present in the ground-truth labels of the test set. In response to our initial findings, the TexTeller authors acknowledged a subset of these issues and temporarily retracted their technical report. However, their defense posits that the officially reported model was trained on filtered data and attributes the observed phenomena to other factors. Our analysis refutes these counterclaims and demonstrates the model's failure to generalize on genuinely novel data. This audit underscores the critical need for rigorous data provenance verification and transparent evaluation protocols to ensure the integrity of academic benchmarking in the field.

Our Reproduced Code: https://github.com/Tex-whistleblower/Data-Leakage-Audit-of-TexTeller



(a): Train–test overlaps on public handwritten expression benchmarks.



(b): HME test examples rendered via the third-party generative renderer *nano-banana*; in the shown samples, PosFormer outputs are garbled.

Figure 1 : Evidence summary. Left: Overlap of benchmark data in TexTeller's training set. Test set contamination rates: For each public benchmark, we plot the percentage of its test images found in TexTeller's training data. The overlap is extremely high indicating pervasive data leakage. Right: Follow-up tests use a third-party generative renderer (*nano-banana*) to argue generalization, yet the shown all PosFormer (SOTA 2024) outputs are garbled under that pipeline; thus, this panel is not a valid evaluation of PosFormer and instead exposes comparability risks and Texteller authors' limited understanding of PosFormer and HMER.

1 Introduction

The field of handwritten mathematical expression recognition (HMER) has seen significant progress, driven by the development of sophisticated models and the availability of public benchmarks. TexTeller Li et al. (2025) is an OCR model that was reported to achieve breakthrough accuracy on handwritten mathematical expression benchmarks (e.g. CROHME Mouchere et al. (2014); Mouchère et al. (2016); Mahdavi et al. (2019), HME100K Yuan et al. (2022)).

However, the validity of such evaluations is predicated on a foundational principle of machine learning: the test data must remain unseen during training. Any violation of this principle, known as data leakage or test set contamination, can lead to drastically inflated performance metrics that reflect memorization rather than true generalization. Recently, concerns were raised regarding the composition of TexTeller's training data, alleging that it contained a substantial portion of the test sets from widely used public benchmarks.

This paper presents a systematic audit to investigate these concerns. We conduct a multi-faceted analysis that moves from establishing the existence of data leakage to demonstrating its direct consequences on model performance and behavior. Our investigation is structured as follows:

- Section 2 provides a quantitative analysis of the overlap between TexTeller's released training data and the test sets of numerous public HMER benchmarks, establishing the scale of the contamination.
- Section 3 demonstrates the direct impact of this leakage on reported performance metrics, highlighting anomalous patterns that are classic signatures of overfitting to a compromised test set.
- Section 4 presents irrefutable qualitative evidence of test-set memorization, a phenomenon we term "error-label memory," where the model reproduces idiosyncratic annotation errors from the test labels.
- Section 5 critically examines the TexTeller authors' counter-arguments and presents further evidence of the model's profound failure to generalize on genuinely novel data.
- Section 6 concludes by synthesizing our findings and offering concrete recommendations for the TexTeller authors and the broader HMER community to prevent such methodological failures and uphold scientific integrity.

Summary and scope. Our analysis yields three concise conclusions.

1. **Data filtering inconsistency.** In email correspondence the authors state that, for handwritten data, the HuggingFace/GitHub release is provided merely as a convenient aggregation and is *not* strictly split into train/val/test; yet later they assert that *TexTeller_en* was trained with a strict isolation of official train/test sets and that the reported metrics are not inflated. This inconsistency leaves open whether filtering actually occurred for the trained model, and—if so—**why the filtered corpus is not released while an unfiltered one is released on Huggingface**.
2. **Evaluation comparability.** The authors indicate that certain *ExpRate* results rely on undisclosed "synonym replacements" (e.g., delimiter substitutions). **Such temporary post-processing is not part of prior HMER baselines, the corresponding code is unavailable, and the practice hampers fair comparison**. In this paper we report *CDM ExpRate* under a transparent, reproducible protocol.
3. **Generalization claims not substantiated.** Despite the technical report claiming >80% *ExpRate* on most benchmarks, *TexTeller_en* performs poorly on genuinely novel inputs: on ten author-provided "Nano Banana" renderings, 8/10 predictions contain recognition errors (3/10 are unrenderable) and only 2/10 are fully correct; performance on MathWriting (including its synthetic 50K subset) is also weak. Moreover, the cited failure of *PosFormer* Guan et al. (2024) on these images stems **from a nonstandard preprocessing mismatch** (RGB photos rather than the binarized 0/1 bitmaps expected by standard HMER pipelines); our re-evaluation with the canonical binarized pipeline does not reproduce such extreme degradation (see Figure 5 and our GitHub).

Throughout, we adopt an evidence-based, neutral tone; quantitative and contested statements are supported by literal citations to the audit report [AuditPDF], the author responses [AuthorResp], and timestamped emails [Email YYYY-MM-DD HH:MM].

2 Public Test Contamination in Training Data

2.1 Experimental Setup

Datasets.

- **CROHME (2014/2016/2019).** Originally online but widely used offline by rasterizing the strokes. The standard training split has 8,836 expressions; the test sets contain 986/1,147/1,199 expressions for 2014/2016/2019 Mouchere et al. (2014); Mouchère et al. (2016); Mahdavi et al. (2019).
- **CROHME 2023.** The 7th CROHME edition adds 3,905 new handwritten equations and introduces three modalities—online, offline, and bi-modal—while inheriting earlier CROHME data. The release provides PNG images, InkML files, and symbol-level label graphs (SymLG) for evaluation Xie et al. (2023)
- **HME100K.** A real-world, large-scale offline dataset (≈100k images) collected from tens of thousands of writers, mainly captured by cameras; official splits report 74,502 training and 24,607 test images Yuan et al. (2022).
- **MathWriting.** The largest online HME dataset to date: about 230k human-written expressions plus about 400k synthetic ones; provided in InkML and readily rasterized for offline HMER Gervais et al. (2025).
- **MNE.** A test set targeting structural complexity with three subsets by nesting depth: N1, N2 drawn from CROHME tests, and N3 collected by authors Guan et al. (2024).

Metric.

- **ExpRate.** The standard CROHME metric—percentage of test expressions whose predicted LaTeX matches the reference exactly (perfect expression-level match).
- **ExpRate@CDM.** An image-level metric designed to reduce LaTeX-string ambiguity by matching detected characters/positions; we report the share of samples with a perfect CDM alignment (CDM=1) Wang et al. (2025). **Note that under compliant practice, image-level CDM ExpRate should be greater than or equal to text-level ExpRate**;

LaTeX Normalization. To compare LaTeX formulas faithfully, we first normalize expressions to remove purely syntactic variation. The audit report specifies a minimal procedure—strip the outer display delimiters and all whitespace—which we adopt as our baseline. Building on this, we use the following canonicalization protocol:

- **Whitespace removal.** Remove all whitespace characters.
- **Delimiter stripping.** Remove outer display-mode delimiters such as \[and \]; we treat the content as a standalone expression.

Overlap matching protocol. To quantify contamination, we compare each benchmark split against the *Tex80M–handwritten* release (1M items; union of `handwritten_online` and `handwritten_nature`). We first canonicalize *both* sides by LaTeX normalization. We then mark a benchmark item as *found in train* if and only if its canonical string appears *exactly* (string equality) at least once among the canonicalized labels in Tex80M–handwritten; duplicates on the training side are ignored. For each split E, we report

$$\text{Found in train } |F| \;=\; \big|\{\, e \in E :\; \text{canon}(e) \in \mathcal{T} \,\}\big|, \qquad \text{Overlap } (\%) \;=\; 100 \times \frac{|F|}{|E|},$$

where $\mathcal{T}$ is the set of canonicalized labels from the training release. This protocol yields an *exact-match*, label-string overlap and intentionally errs on the conservative side by avoiding any semantic or renderer-dependent equivalences.

2.2 Overlap Rates with Benchmarks

Using the methodology described in the audit report, we computed the fraction of each benchmark's test set that was present in TexTeller's released training data on HuggingFace (the "handwritten_online" + "handwritten_nature" subsets of Tex80M). The results, summarized in Figure 1a and Table 1, reveal pervasive contamination. For instance, **94.22%** of the CROHME 2023 official test set images (2167 out of 2300) were found verbatim in TexTeller's training data Similarly, the older CROHME 2014/2016/2019 test sets show

Table 1: Central train–test overlaps with the Tex80M–handwritten release. For each public benchmark split, we canonicalize its reference LaTeX (whitespace removal and outer-delimiter stripping only) and test for exact string membership in the canonicalized labels of *Tex80M–handwritten* (1M; `handwritten_online` ∪ `handwritten_nature`). We report the number of evaluation samples (*Total*), how many are found at least once in the training release (*Found in train*), and the resulting *Overlap (%)*.

Dataset	Split	Total	Found in train	Overlap (%)
CROHME				
	CROHME 2014 test	986	913	92.60
	CROHME 2016 test	1,147	1,031	89.89
	CROHME 2019 test	1,199	1,056	88.07
	CROHME train	8,834	8,039	91.00
CROHME-2023				
	train	12,204	12,168	99.71
	val	555	531	95.68
	test	2,300	2,167	94.22
HME100K				
	train	74,502	74,490	99.98
	test	24,607	24,607	100.00
MathWriting				
	train	229,836	166,443	72.42
	synthetic train	395,711	228,887	57.84
	val	15,670	11,744	74.95
	test	7,644	6,634	86.79
UniMER				
	HWE test	6,332	6,000	94.76
MNE				
	N1 test	1,875	1,624	86.61
	N2 test	304	240	78.95
	N3 test	1,464	379	25.89

overlap rates of **92.6%**, **89.9%**, and **88.1%**, respectively The HME100K dataset's test partition (24,607 images) was *entirely* included in training (100.00% overlap) Other datasets exhibit the same pattern: e.g. UniMER-HWE Wang et al. (2024) test 94.8%, MNE-N1 test 86.6%, MNE-N2 test 79.0% overlap Even MathWriting, a large synthetic-and-handwritten formula dataset, shows an 86.8% overlap in its test set The only partial exception is the MNE-N3 test, with about 25.9% of its images found in training —nonetheless a nonzero overlap. These rates indicate that TexTeller's training set *almost completely covers the content of most public test benchmarks*, including the entirety of some test sets. In many cases, it appears that TexTeller's 80M data was augmented by wholesale incorporation of widely-used evaluation sets.

Notably, TexTeller's training set also contained large portions of the official *training* and *validation* splits of these benchmarks. For example, 91.0% of images from the CROHME 2014–2019 training sets were found, as well as 95.68% of CROHME 2023 validation images This suggests the data collection indiscriminately scooped up entire datasets. While using external training data is not inherently problematic, including an evaluation's **test set** in one's training data fundamentally undermines the validity of any reported result on that evaluation. In TexTeller's case, the overlap is so complete that the model could essentially *memorize the answers* to the test benchmarks, as we explore in the next section.

It is important to establish a baseline. Using the same canonicalization protocol as Section 2.1, we verified that **under normal conditions exact-string overlaps are minimal**. The extreme overlaps (88–100%) reported earlier are computed *against the authors'* **Tex80M–handwritten** *subset* (approximately 1M handwritten formulas). For comparison, Table 2 shows overlaps when the same test splits are searched in an unrelated corpus, the MathWriting training set (230k, Google 2024). In this baseline, only 1–4% of items overlap (e.g., just 16 of 1,199 CROHME 2019 test images, 1.33%), plausibly due to a few identical expressions or incidental string matches. This stark contrast indicates that the 88–100% overlaps observed for *Tex80M–handwritten* are not random noise but consistent with bulk inclusion of those test sets in the training release.

Table 2 : Baseline overlap rates against an unrelated dataset (MathWriting). Each entry reports how many items from external splits are found in the MathWriting handwritten training set (230k). Only 1–4% of formula overlapped, compared with TexTeller's overlap rate of 90%.

Dataset	Split	Total	Found in MW train	Overlap (%)
CROHME				
	CROHME 2014 test	986	42	4.26
	CROHME 2016 test	1,147	19	1.66
	CROHME 2019 test	1,199	16	1.33
	CROHME train	8,834	692	7.83
HME100K				
	train	74,502	621	0.83
	test	24,607	77	0.31

2.3 Visual confirmation of overlaps

After computing string-level overlaps in §2.1, we further verify that many matches correspond to *image-level* duplicates or near-duplicates. Our procedure is as follows: (i) canonicalize all LaTeX labels on both sides using the Section 2.1 rules (whitespace removal and outer-delimiter stripping only); (ii) take the exact string intersection between each benchmark split and the *Tex80M–handwritten* release; (iii) for each intersecting label, retrieve one representative image from the benchmark and one from *Tex80M–handwritten*; (iv) visualize the pair side-by-side without any alignment beyond uniform resizing for layout. Figure 2 shows representative pairs for four test sets—HME100K, MathWriting, CROHME 2019, and UniMER–HWE—where the **left** column displays the benchmark test image and the **right** column shows the counterpart found in *Tex80M–handwritten*. Across datasets we consistently observe that the *Tex80M* images are either identical copies or differ only by trivial transforms (e.g., aspect-ratio stretching or mild noise changes), confirming that the high label-string overlaps translate into content-level contamination.

2.4 Author responses

First response (Email 2025-08-22 16:42). After we shared overlap counts and paired visual matches, the authors updated the GitHub/HuggingFace pages and wrote that, for the handwritten part of the corpus, the release is intended merely as a convenient aggregation for the community and is *not* strictly partitioned into train/val/test. They further stated that, given the fragmentation of handwritten-formula datasets, they would no longer maintain strict splits themselves and welcomed third parties to release a dedicated dataset for HMER [Email 2025-08-22 16:42].

Second response (Email 2025-08-31 16:37). In a subsequent email, the authors asserted that the training of *TexTeller_ en strictly* isolated the official train/test sets and that the reported metrics "are neither contradictory nor inflated" in practical scenarios [Email 2025-08-31 16:37].

Inconsistencies and open questions. The two claims are difficult to reconcile: the public GitHub/HuggingFace release is acknowledged to include validation/test material and is not split, yet the trained model is said to use a strictly filtered corpus with train/test isolation. To date, *no* filtering code, checksums, or filtered corpus has been released; only the unfiltered aggregation is public. This blocks independent verification and leaves the extent of training contamination unknown.

Implications for reproducibility. Without access to the filtering pipeline or the filtered corpus, the community can evaluate only the public, unfiltered release—the same data our overlap analysis shows to be contaminated. We therefore adopt a transparent, reproducible protocol and clearly separate (i) overlaps measured on the public release from (ii) unverified claims about a private filtered variant.

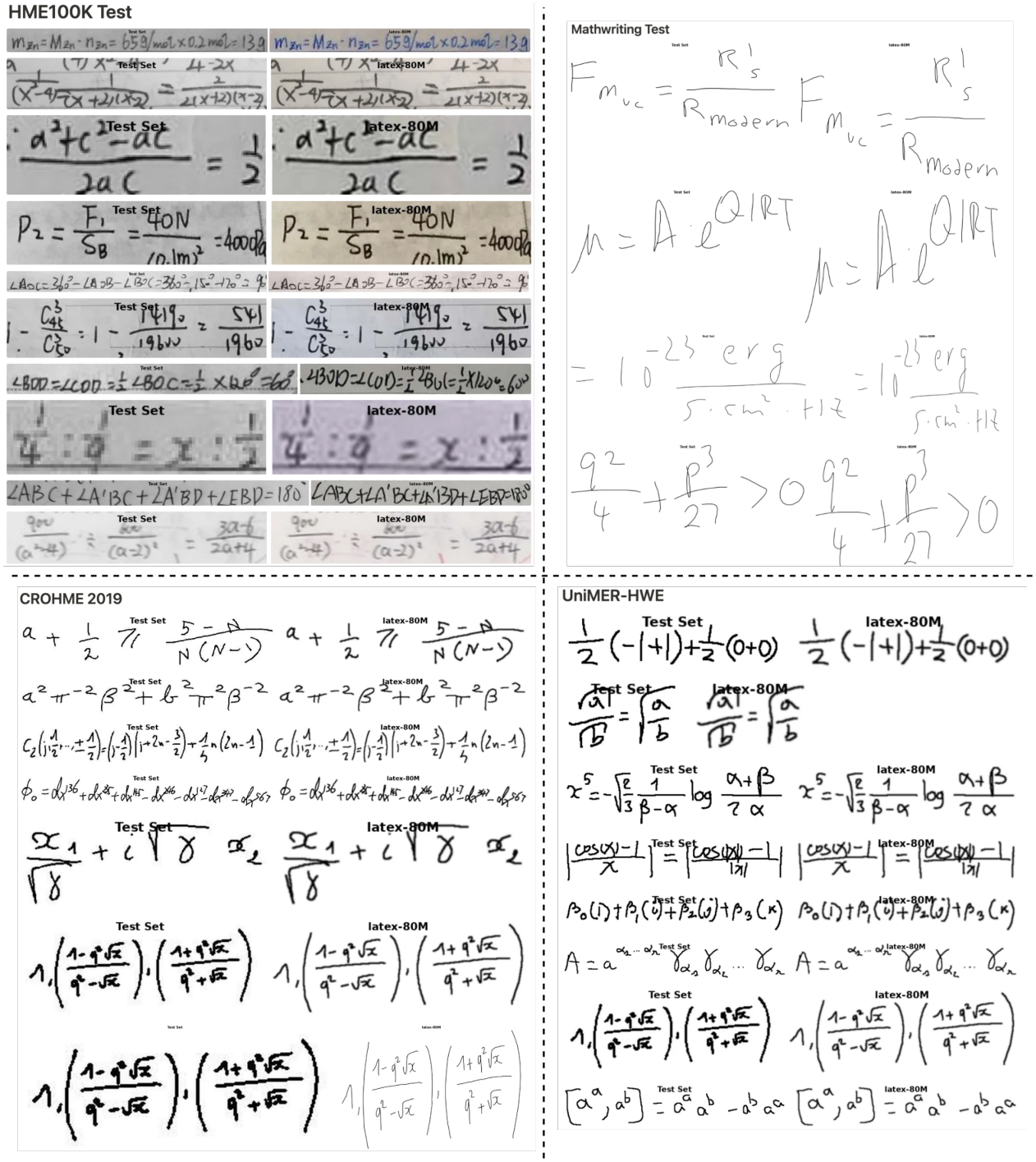


Figure 2 : Side-by-side visualization of canonical-label overlaps. For each dataset block, the *left* panel shows examples from the official *test set*, and the *right* panel shows the matched images found in the *Tex80M–handwritten* release. In most cases, the *Tex80M* counterparts appear identical or differ only by simple geometric/intensity augmentations (e.g., stretching, contrast jitter), indicating duplicated or trivially augmented versions of benchmark items rather than genuinely novel samples.

3 Impact on Performance

The contamination of test data in training provides a compelling explanation for TexTeller's extraordinarily high reported accuracy. In standard practice, a model's accuracy on a held-out test set should be slightly lower than or at best on par with its validation performance (since the model may see the validation set during development, but not the test set).

Table 3 : TexTeller_en on public splits grouped by observed phenomena. (Evaluated on CDM complete-match %) CDM values are reproduced from the audit table; ExpRate values are from the TexTeller technical report. "Previous SOTA" figures are approximate and not normalized to our unified protocol. The report does not provide per-level ExpRate for some benchmarks (marked "–"). The red numbers indicate ExpRate > CDM ExpRate. Boldface highlights the side that characterizes the phenomenon.

Dataset	Split	ExpRate (%)	CDM ExpRate (%)	Previous SOTA
Phenomenon 1: Train Accuracy ≈ Test Accuracy				
CROHME	Train	–	96.38	–
	2014 Test	88.0	95.64	∼65%
	2016 Test	85.9	92.68	∼65%
	2019 Test	85.8	92.08	∼65%
MNE	N1	–	93.49	∼65%
	N2	–	90.46	∼60%
	N3	–	88.18	∼50%
HME100K	Train	–	90.25	–
	Test	90.7	87.45	∼70%
Phenomenon 2: Test/Validation > Train (validation higher)				
CROHME2023	Validation	–	**96.21**	–
	Test	61.8	95.52	–
	Train	–	87.96	–
Phenomenon 3: Overall low performance (CDM ExpRate $\lesssim$ 70%)				
MathWriting	Train (20K)	–	67.53	–
	Synthetic (50K)	–	52.54	–
	Validation	–	58.23	–
	Test	81.0	61.92	–

Split-level accuracies invert the usual train≥val≥test pattern. Under proper isolation, a model that repeatedly sees the training set typically attains *higher* training accuracy than on val/test; here we observe the opposite or near-equality. For *CROHME 2023*, TexTeller's CDM rises on held-out splits (val 96.21%, test 95.52%) while *training* is lower (87.96%); *MathWriting* likewise shows test 77.29% > val 68.88%; and *CROHME 2014/2016/2019* test 99.39/98.78/97.58% nearly matches CROHME-train 99.18% [AuditPDF p.5]. These split-level patterns are consistent with the audit's measured train–test overlaps on the same benchmarks (e.g., CROHME, HME100K) [AuditPDF p.2] and warrant caution when interpreting unusually high headline scores.

Replacement-based ExpRate and mixed pipelines undermine metric comparability. In email, the authors acknowledge applying *replacements* when computing ExpRate to reconcile dictionary differences, which deviates from the conventional ExpRate and should not be mixed with historical reports in the same table [Email 2025-08-31 16:37]. Under a *unified* tokenizer/dictionary and a single public synonym layer applied equally to predictions and references, the *CDM complete-match rate* ("CDM ExpRate") should not undercut text-only ExpRate; if it does, that flags a pipeline or rendering mismatch rather than genuine recognition gains [Email 2025-08-25 13:10]. Yet two cases in the attachments show exactly such reversals: on *HME100K*, the tech-report ExpRate is 90.7 while our reproduced CDM complete-match is 87.45 [Email 2025-08-25 13:10]; on *MathWriting*, the audit table shows test CDM complete-match 61.92% whereas the tech-report ExpRate cited in the thread is higher (reported as 81.0%) [AuditPDF p.5; Email 2025-08-25 13:10]. These mismatches

are plausibly explained by nonstandard replacements and mixed evaluation pipelines; notably, the specific replacement rules and code are not provided in the attachments, reinforcing our recommendation to re-score *all* systems under one unified protocol. Besides, using the replacement method to calculate ExpRate would result in an **unfair comparison** with previous methods.

Generalization is weak on genuinely novel data despite strong claims. Although the response argues broad robustness, TexTeller's accuracy drops sharply off-distribution: on *MathWriting*, a 20k subset drawn from its own training distribution yields 67.53% (CDM), while a 50k synthetic subset falls to 52.54%; the official test sits in between (CDM 61.92%) [AuditPDF p.5]. This pattern—strong on familiar data, weak on novel synthetic samples—is consistent with overlap-driven familiarity rather than true generalization, and further motivates a unified, fully documented evaluation on truly unseen splits.

4 Error-Label Memory Analysis

Why erroneous labels matter. One of the most compelling pieces of evidence for test contamination is TexTeller's tendency to reproduce *labeling mistakes* present in the test data. If a test sample's ground-truth formula has an error (due to annotation mistakes), a well-generalized model might actually output a *different* formula – perhaps the correct interpretation of the input – instead of mimicking the error. In TexTeller's case, however, we found multiple instances where the model's output exactly matched the flawed label, indicating it had likely memorized that label from training data.

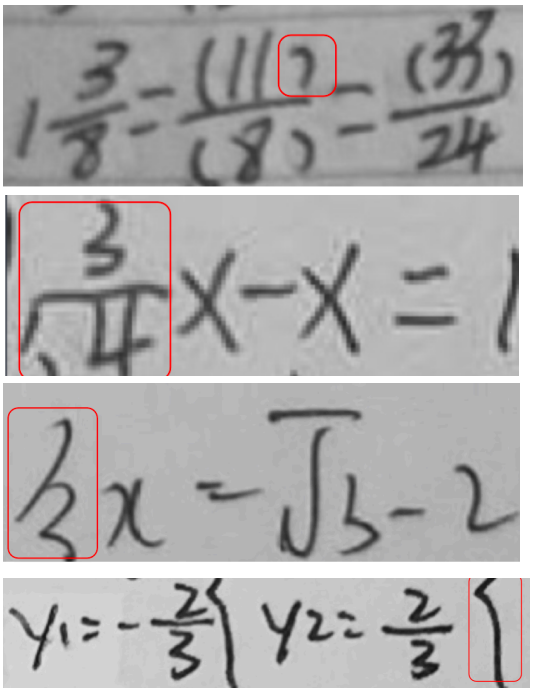


Image ID: test_28698
Ground Truth: 1 \frac{3}{8} = \frac{(117)}{(8)} = \frac{(33)}{24}
Prediction: 1 \frac{3}{8} = \frac{(117)}{(8)} = \frac{(33)}{24}
Analysis: Both misread "11)" as "117".

Image ID: test_23723
Ground Truth: _{1、} \frac{3}{4}x - x = 1
Prediction: _{1、} \frac{3}{4}x - x = 1
Analysis: Both misread denominator "4" as subscript _{1、}.

Image ID: test_23003
Ground Truth: ^{1}3x=\sqrt{3}-2
Prediction: ^{1}3x=\sqrt{3}-2
Analysis: Both misread "令" as superscript ^{1}3.

Image ID: test_28607
Ground Truth: y_{1}=- \frac{2}{3}[y_{2}= \frac{2}{3}[
Prediction: y_{1}=- \frac{2}{3}[y_{2}= \frac{2}{3}[
Analysis: Both end with "[`" instead of "]".

Figure 3 : Error–label memorization examples included in our first email to the authors. For four HME100K test images, *TexTeller_ en* reproduces the annotation mistakes in the ground-truth labels *exactly*. Red boxes mark the loci of the errors. These exact GT–Pred matches provide direct evidence of error–label memory on HME100K.

Evidence from Concrete Cases. We examined HME100K test samples where the official labels contain mistakes or ambiguous glyphs. As is shown in Figure 3, both the human labeler and TexTeller misinterpreted a symbol in the same incorrect way. From the model's perspective, this results in an output identical to the (wrong) label, which could be mistaken for memorization.

Author Communications and Explanations. In email correspondence, the Texteller authors (i) acknowledged the **3/4** matches and retracted their arXiv technical report for revision. After a week, they proposed two explanations: *Ambiguity coincidence* (humans and the model independently make the same misread on ambiguous handwriting), and *implicit data leakage* arising from large-scale data collection (e.g., fragmented long-formulas across train/test, or synthetic-composition artifacts that couple train/test token patterns).

These explanations do not contradict our finding that exact wrong-label matches occur. After our email, the Texteller authors acknowledged that the Hugging Face release included the test set but claimed they manually filtered validation/test data during training; however, they have not provided the filtered data, a standardized definition and computation of ExpRate, or the corresponding evaluation results and pipline.

5 Counterclaims and Additional Evidence

In light of our findings, the TexTeller authors have presented several counterarguments and additional pieces of evidence to support the integrity and capability of their model. In this section, we present these points impartially and discuss their implications.

Further Error–Label Memory Cases in Author Responses. Despite the authors' email claim that they screened the first 1,000 HME100K test samples to identify instances where the ground-truth label was erroneous but *TexTeller_en* produced the correct formula, our re-evaluation of the author-provided cases with the released *TexTeller_en* model reaches a different conclusion. As illustrated in Figure 4, we still observe multiple *exact* matches between the model outputs and the erroneous labels on the selected HME100K examples. These concordant mistakes persist under the authors' decoding setup and constitute additional evidence of error–label memorization rather than systematic correction of label noise.

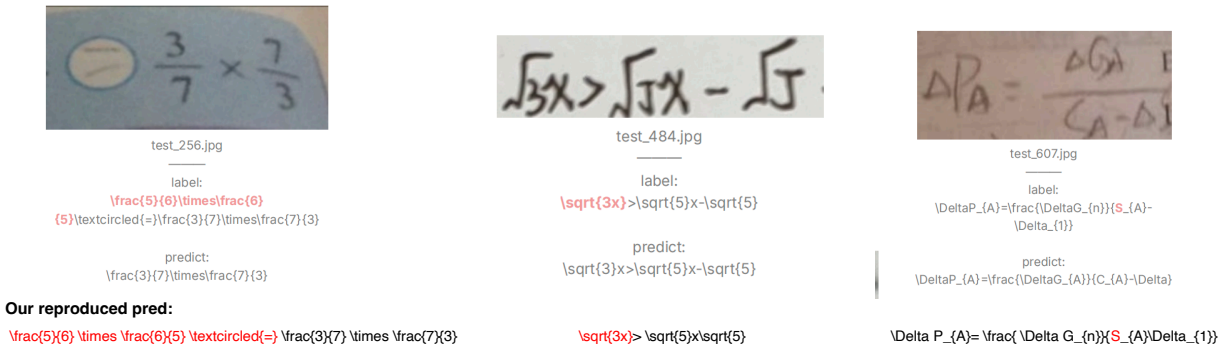


Figure 4: Author-provided error–label cases from HME100K (left to right: `test_256`, `test_484`, `test_607`). Each panel shows the test image, the official label, and the authors' *TexTeller_en* prediction; red markup highlights erroneous tokens in the label. Re-running the released *TexTeller_en* the same wrong outputs (bottom, red)—i.e., exact matches to the erroneous labels—corroborating error–label memorization.

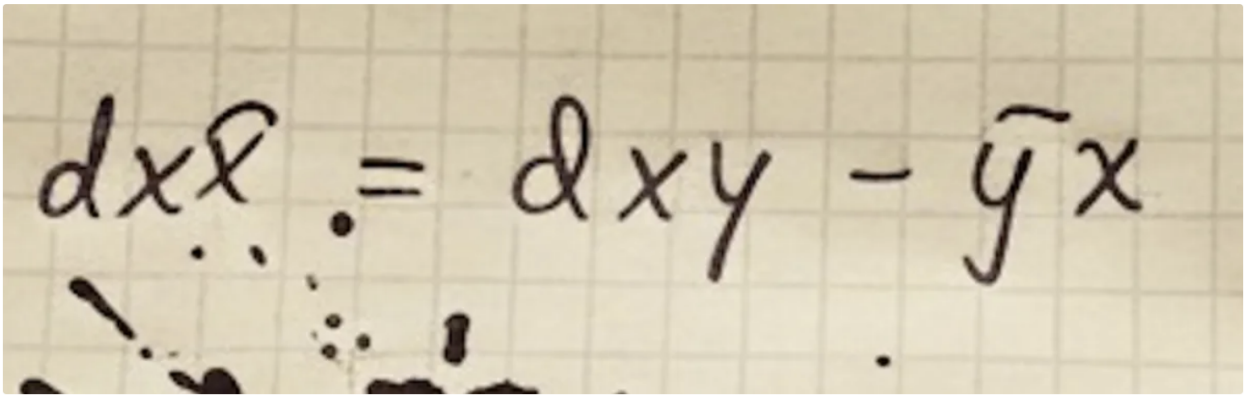
To argue that *TexTeller_en* generalizes beyond public benchmarks, the authors compared it with *PosFormer* Guan et al. (2024) (an open-source HMER model reported to be **SOTA** on CROHME, MNE, and HME100K in 2024) by generating ≈ 10 new handwritten-style images using "Gemini 2.5 Flash Image (Nano Banana)". As is shown in Figure 5, *TexTeller_en* often produced coherent LaTeX (e.g., $dx\overline{x} = dxy - \overline{y}x$), whereas *PosFormer* outputs were largely nonsensical (e.g., repeated y-tokens), which the authors cite as evidence of superior generalization rather than overfitting.

Gaps in the TexTeller Authors' Discussion of HMER However, this comparison is methodologically weak: standard HMER practice—and *PosFormer*'s pipeline—expects binarized (0/1) bitmap inputs rather than RGB photos, so the observed failures of PosFormer likely reflect a preprocessing mismatch. Thus, the presented evidence neither establishes broad generalization nor refutes error–label memorization, and it reveals a limited

understanding of prior HMER methods; rigorous, binarized, and sufficiently large evaluations for comparison are required.

Standardized PosFormer re-evaluation. To rule out preprocessing confounds, we re-ran *PosFormer* using its official binarized (0/1) preprocessing and decoding pipeline. With a public checkpoint trained *only* on the 8,836-image CROHME train set, the model does not collapse into repeated tokens in all cases; its outputs are syntactically valid and legible on the Nano–Banana renderings (see Figure 6 for examples). This indicates that the garbling observed in the authors' comparison is attributable to a pipeline mismatch rather than an inherent failure of *PosFormer*.

Performance of TexTeller_en on 10 new handwritten-style images generated by Nano Banana. We evaluated the released *TexTeller_en* on ten novel handwritten-style renderings produced by the "Nano Banana" engine shared by the authors. As is shown in Figure 6, despite reporting ExpRates above 80% on most of the public HMER datasets in the technical report, the model performs poorly on these unseen inputs: 8/10 predictions contain recognition errors (including 3/10 that yield LaTeX strings that do not render), and only 2/10 are fully correct. This gap, together with the weak results on MathWriting noted earlier, indicates limited robustness to style shifts outside the training distribution.



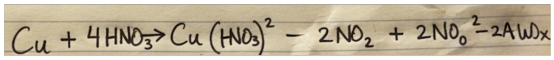
posformer:

```
\lim \limits _ { y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\lim \limits _ { y \rightarrow y , y , y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y , y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y , y , y , y , y , y , y | y , y , y , y , y , y , y , y ,
y , y , y , y \rightarrow y \rightarrow y , y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y
\rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow p \rightarrow p \rightarrow
```

texteller_en:

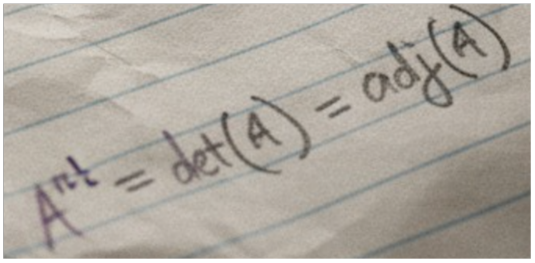
```
dx \overline{x}=dxy- \overline{y}x
```

Figure 5: Author-provided "Nano Banana" rendering used to compare *TexTeller_en* and *PosFormer*. However, this outcome is confounded by a evaluation mismatch with standard HMER practice—*PosFormer* expects binarized (0/1) bitmap inputs rather than RGB photos—so the example should not be taken as evidence of broader generalization.



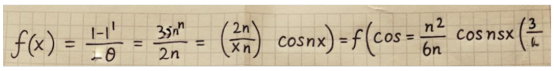
1) **TexTeller_en Prediction:**
Cu+4HNO_{3} \rightarrow Cu(HNO_{3})^{2}-2NO_{2}+2NO_{0}^{2}-2A(O_{x}
Analysis:
Terminal "AlO_x" is mis-OCR'd as "A(O"
the letter "l" ande missing causing unmatched brackets.
Our Reproduced PosFormer Prediction:
C_{u}+4N_{3}>C_{u}(1N_{3})^{2}-2N_{0}-2N_{2}j_{i}^{-2}-\sumAA_{0}^{\cdots+\infty}
Renderable: (Texteller_en pred → LaTeX)

$$Cu + 4HNO_3 \rightarrow Cu(HNO_3)^2 - 2NO_2 + 2NO_0^2 - 2A(O_x$$

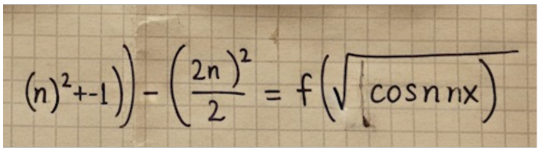


2) **TexTeller_en Prediction:**
A^{ \pi t}=det \left(A \right)= \alpha \dot{ \alpha}j \left(A \right)
Analysis:
"adj(A)" is misrecognized as the token sequence "α·α j (A)"
the superscript "T" in \(A^T\) is misread as "π t".
Our Reproduced PosFormer Prediction:
\sum\limits_{n}^{s_{i}}=x^{\sum\limits_{i}}{\frac{s_{i}}{n,\ldots)^{2}}{\cdots+s_{i}}\cdot{\frac{n,j\prime\sin{j\prime\sin^{\frac{j}{1}}}}
Renderable: (Texteller_en pred → LaTeX)

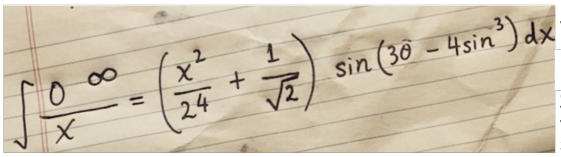
$$A^{\pi t} = det\,(A) = \alpha\dot{\alpha}j\,(A)$$



3) **TexTeller_en Prediction:**
f \left(x \right)= \frac{1-1^{1}}{ \theta}= \frac{3in^{n}}{2n}= \left(\frac{2n}{xn} \right) \cos nx
=f \left(\cos= \frac{n^{2}}{6n} \cos nsx \left(\frac{3}{6} \right. \right.
Analysis:
The expression truncates after "\cos nx"
extra right } appear after \(n^{2}\)
the closing "\right." is duplicated yielding unbalanced delimiters.
Unrenderable : (Texteller_en pred → LaTeX)
Our Reproduced PosFormer Prediction:
f(x)=\frac{1-1}^{n}}{2n}=(\frac{2n}{xn})=f(\cosx)=\frac{n^{2}}{6n}\cos(\frac{3}{k}

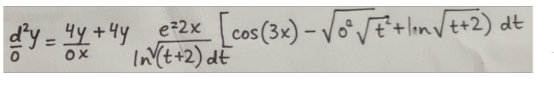


4) **TexTeller_en Prediction:**
\left. \left(n \right)^{2}+-1 \right) \right)- \left(\frac{2n \right)^{2}}{2}=f \left(\sqrt{ \cos nnx} \right)
Analysis:
Counts of "\left" and "\right" do not match
an extra "\right)" occurs outside the radical inconsistent with the source
Unrenderable: (Texteller_en pred → LaTeX)
Our Reproduced PosFormer Prediction:
(n)^{2}+1)^{i}{\frac{2n}^{2}}{(\frac{1}{i-1}}=f(\frac{1-1}{\frac{1}{i\cdot!}{2}\sin(nx)}



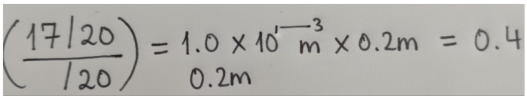
5) **TexTeller_en Prediction:**
\int \frac{0 - \infty}{x}= \left(\frac{x^{2}}{24}+ \frac{1}{ \sqrt{2}} \right) \sin \left(3 \theta-4 \sin^{3} \right) dx
Analysis:
The integral limits were misread: "0 ∞" was recognized as "0 - ∞"
replacing the limit notation with a minus.
Our Reproduced PosFormer Prediction:
\int\limits_{i}^{0}\frac{x^{2}}{x}=(\frac{1}{2^{4}}+\frac{1}{\sqrt{2}})\sin(3\theta-4\sin^{3})dx}
Renderable: (Texteller_en pred → LaTeX)

$$\int \frac{0-\infty}{x} = \left(\frac{x^2}{24} + \frac{1}{\sqrt{2}}\right) \sin\left(3\theta - 4\sin^3\right) dx$$



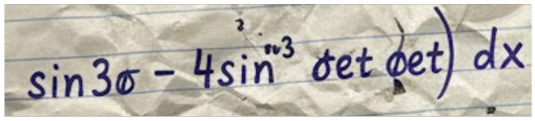
6) **TexTeller_en Prediction:**
\frac{d^{2}y}{dx^{2}}= \frac{4y}{ox}+4y \frac{e^{2}2x}{ \ln^{2}
\left(t+2 \right) d t} \left[\cos \left(3x \right)- \sqrt{o^{2} \sqrt{t}^{2}}+ \ln \sqrt{t+2} \right) dt
Analysis:
The denominator "ox" is ambiguous (should be "0" or "o")
a radical was parsed as a fraction around "e^{2}2x"
"o^{2}" which should just be o^{o} or o^{a}.
Our Reproduced PosFormer Prediction:
\frac{d^{0}y=\frac{d^{2}}{4x}+4y+\frac{e^{2x}{(t+2)}dt}{\cos(3x)-\sqrt{0^{a}\sqrt{t^{2}+\lim\sqrt{t+2}dt
Renderable: (Texteller_en pred → LaTeX)

$$\frac{d^2y}{dx^2} = \frac{4y}{ox} + 4y\frac{e^2 2x}{\ln^2(t+2)\,dt}\left[\cos(3x) - \sqrt{o^2\sqrt{t^2}} + \ln\sqrt{t+2}\right) dt$$

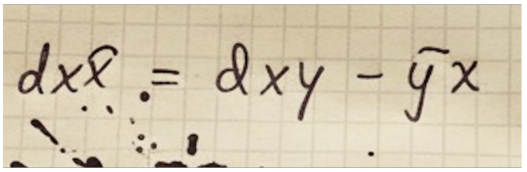


7) **TexTeller_en Prediction:**
\left(\frac{17/20}{120} \right)=1.0 \times 10^{-3}m \times 0.2m=0.4
Analysis:
The lower multiplicative factor "0.2 m" from the layout is missing.
Our Reproduced PosFormer Prediction:
{\frac{17/20}{120}}=1.0\times10^{1-2m^{3}\times0.2m=0.4}
Renderable: (Texteller_en pred → LaTeX)

$$\left(\frac{17/20}{120}\right) = 1.0 \times 10^{-3} m \times 0.2m = 0.4$$

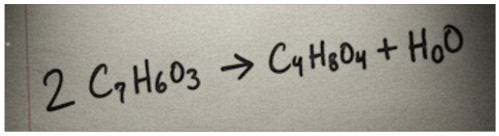


8) **TexTeller_en Prediction:**
\sin 3 \pi-4 \sin^{-3}det \phi et \right) dx
Analysis:
The symbol "\sigma" was misread as "\pi"
the closing "\right)" lacks a matching "\left" making the LaTeX unrenderable
Unrenderable: (Texteller_en pred → LaTeX)
Our Reproduced PosFormer Prediction:
\lim\limits_{n\rightarrow0}-4\cdot\frac{3}{3}\cdot\lim\limits_{m3}8ek\phi_{n}\phi_{k})^{2}dx



9) **TexTeller_en Prediction:**
dx \overline{x}=dxy- \overline{y}x
Analysis:
TexTeller_en Recognize **Correct**
Our Reproduced PosFormer Prediction:
\lim\limits_{x\rightarrow0}=dx_{0}\cdoty-y\timesyx
Renderable: (Texteller_en pred → LaTeX)

$$dx\overline{x} = dxy - \overline{y}x$$



10) **TexTeller_en Prediction:**
2C_{9}H_{6}O_{3} \rightarrow C_{9}H_{8}O_{4}+H_{0}O
Analysis:
The "9" is ambiguous (may be "7" or "9")
TexTeller_en Recognize **Correct**
Our Reproduced PosFormer Prediction:
\lim\limits_{\frac{\sum\limits_{3}\theta_{4}}{4}+\ldots+\lim\limits_{\cdots+\lim\limits_{\frac{1}{4}\prime\sin^{\frac{1}{4}+\beta_{4}}}=\beta
Renderable: (Texteller_en pred → LaTeX)

$$2C_9H_6O_3 \rightarrow C_9H_8O_4 + H_0O$$

Figure 6 : Qualitative audit of *TexTeller_en* on author-provided "Nano Banana" renderings. The figure annotates each sample with the predicted LaTeX and a minimal diagnosis; Overall, across the ten cases, 8/10 are wrong (3/10 unrenderable) and only 2/10 are correct, contrasting with the $> 80\%$ ExpRates reported on most of the public benchmarks. Besides, we reproduced the PosFormer prediction in a standard evaluation, which shows a normal recognized result.

6 Conclusion

In summary, TexTeller's remarkable benchmark results appear to stem largely from test data leakage into its training set, rather than true generalization. We strongly recommend that TexTeller be re-evaluated under a single, transparent evaluation protocol that strictly separates training and test data. This case highlights the necessity of rigorous data provenance checks and strict dataset separation to ensure that benchmark results are fair and credible in the field of HMER.

This case highlights a critical vulnerability in the current machine learning research ecosystem and underscores the need for greater diligence in data management and evaluation. To address the specific issues raised and to help strengthen the integrity of future research in the field, we offer the following recommendations.

Recommendations for the TexTeller Authors. To restore confidence and enable independent verification, we recommend three concrete actions:

- **Release the filtering pipeline and the actually filtered corpus.** Publish the exact scripts/configs used to isolate training from validation/test, together with the *resulting filtered snapshot* (or a reproducible download recipe), and per-split overlap/deduplication reports against CROHME 14/16/19, HME100K, MathWriting, and UniMER-HWE. The current situation—an unfiltered HuggingFace/GitHub aggregation being public while the alleged filtered corpus remains private—precludes verification of the claimed low-overlap, no-duplicate training set.
- **Standardize ExpRate and ensure comparability.** Release the token-level replacement list used in evaluation (e.g., `\left (` $\rightarrow$ "(") and any other mappings), the full evaluation code, and per-split prediction files. Report both (i) canonical, replacement-free scores and (ii) "with-replacements" scores, and reconcile discrepancies such as MathWriting (81.0% *ExpRate* vs. 61.92% *CDM ExpRate*; Table 3). Note that under compliant practice, image-level *CDM ExpRate* should be *greater than or equal to* text-level *ExpRate*; any reversal warrants explicit justification.
- **Reassess generalization under the standard pipeline.** Re-run the Nano–Banana study using the standard binarized input pipeline and release the images and scripts. Compare against *PosFormer* under its official preprocessing; our re-run with a checkpoint trained only on 8,836 CROHME-train images yields non-garbled outputs (Figure 6), contradicting the original, non-standard comparison. In addition, the authors' own ten Nano–Banana cases show 8/10 incorrect predictions (with 3/10 unrenderable), and new HME100K examples continue to exhibit error–label memory—both indicating limited generalization of *TexTeller_en*.

Pending these steps, claims of state-of-the-art accuracy and broad generalization should be treated as provisional.

References

Philippe Gervais, Anastasiia Fadeeva, and Andrii Maksai. Mathwriting: A dataset for handwritten mathematical expression recognition. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5459–5469, 2025.

Tongkun Guan, Chengyu Lin, Wei Shen, and Xiaokang Yang. Posformer: recognizing complex handwritten mathematical expression with position forest transformer. In *European Conference on Computer Vision*, pp. 130–147. Springer, 2024.

Haoyang Li, Jiaqing Li, Jialun Cao, Zongyuan Yang, and Yongping Xiong. Towards scalable training for handwritten mathematical expression recognition. *arXiv preprint arXiv:2508.09220*, 2025.

Mahshad Mahdavi, Richard Zanibbi, Harold Mouchere, Christian Viard-Gaudin, and Utpal Garain. Icdar 2019 crohme+ tfd: Competition on recognition of handwritten mathematical expressions and typeset formula detection. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1533–1538. IEEE, 2019.

Harold Mouchere, Christian Viard-Gaudin, Richard Zanibbi, and Utpal Garain. Icfhr 2014 competition on recognition of on-line handwritten mathematical expressions (crohme 2014). In *2014 14th international conference on frontiers in handwriting recognition*, pp. 791–796. IEEE, 2014.

Harold Mouchère, Christian Viard-Gaudin, Richard Zanibbi, and Utpal Garain. Icfhr2016 crohme: Competition on recognition of online handwritten mathematical expressions. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 607–612. IEEE, 2016.

Bin Wang, Zhuangcheng Gu, Guang Liang, Chao Xu, Bo Zhang, Botian Shi, and Conghui He. Unimernet: A universal network for real-world mathematical expression recognition. *arXiv preprint arXiv:2404.15254*, 2024.

Bin Wang, Fan Wu, Linke Ouyang, Zhuangcheng Gu, Rui Zhang, Renqiu Xia, Botian Shi, Bo Zhang, and Conghui He. Image over text: Transforming formula recognition evaluation with character detection matching. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19681–19690, 2025.

Yejing Xie, Harold Mouchère, Foteini Simistira Liwicki, Sumit Rakesh, Rajkumar Saini, Masaki Nakagawa, Cuong Tuan Nguyen, and Thanh-Nghia Truong. Icdar 2023 crohme: Competition on recognition of handwritten mathematical expressions. In *International Conference on Document Analysis and Recognition*, pp. 553–565. Springer, 2023.

Ye Yuan, Xiao Liu, Wondimu Dikubab, Hui Liu, Zhilong Ji, Zhongqin Wu, and Xiang Bai. Syntax-aware network for handwritten mathematical expression recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4553–4562, 2022.