Image Over Text: Transforming Formula Recognition Evaluation with Character Detection Matching

Supplementary Material

8. User Preference Evaluation Analysis

To provide a more intuitive and clear analysis of the credibility of CDM, we supplement the content in Section 5.2 with a detailed examination of user preferences for CDM and BLEU metrics under different conditions.

To assess the reliability of CDM, we design an annotation interface as shown in Figure 5. Given the ground truth rendered image and the model's predicted rendered image for various samples, annotators are asked to assign an appropriate score. Score A and Score B correspond to the BLEU and CDM scores of the prediction results, but the order is randomized so that users do not know which score corresponds to which metric. Users make their choice based on their intuitive judgment from four options.

A total of 1008 samples are scored, and the results are categorized into four scenarios. We provide a detailed and clear analysis of user preferences for CDM and BLEU metrics in each scenario, as illustrated in Figure 6:

CDM is better (64%): In this scenario, examples include Case 1 and Case 2. In Case 1, the prediction result is 100% correct, with a CDM score of 1 and a BLEU score of 0. Users directly chose the CDM score. In Case 2, the prediction result is mostly correct, but the BLEU score is significantly lower than expected, leading users to prefer the CDM score.

Both scores are equally good (32%): Examples in this scenario include cases 3 and 4, where the CDM and BLEU scores are relatively close, both reflecting the proportion of model prediction errors in an accurate and intuitive manner.

BLEU is better (3%): In Case 5, due to different token representations of "BF", BLEU detects inconsistencies, while CDM considers BF and $\mathfrak{BF}$ as the same token.

Neither score is good (1%): In Case 6, although the two formulas contain different tokens, `"\mathcal{E}"` and `"\varepsilon"`, they render similar images ($\mathcal{E}$ and ε). Both CDM and BLEU fail in this case.

CDM is reliable in 96% of cases. The remaining 4% are due to LaTeX issues, which will be optimized in future versions, with minimal impact on the overall evaluation.

8.1. Latex Rendering and Syntax Errors

CDM relies on normalizing LaTeX source code and rendering images. Therefore, code that cannot be rendered or contains syntax errors (which cannot be normalized) will result in computation failures. For example, the expression `"z = \left( \begin{array}{cc} x \\ y"` is a failure case due to a missing `"\end{array}"`, leading to rendering failure. For these cases, CDM assigns a score of 0. Although CDM cannot directly handle them, this approach is reasonable and aligns well with human perception.

The number of LaTeX rendering and syntax errors depends on the quality of the model's prediction. Among the four models, Pix2tex, Texify, Mathpix, and UniMERNet, the proportion of LaTeX rendering and syntax errors in the predicted results on the UniMER-Test is 13.83%, 5.03%, 2.38%, and 1.05%, respectively.

8.2. Rendering Types Affecting Token Consistency

CDM defines characters without considering rendering styles. However, different rendering styles can produce visually distinct results, potentially causing different tokens to render into nearly identical characters(Figure 6 Case6), or same tokens to render into different characters(Figure 6 Case5). Similar situations include `"G"` and `"\mathcal { G }"`, `"\mathcal { X }"` and `"\mathfrac { X }"`, whose rendering effects are $G, \mathcal{G}, \mathcal{X}, \mathfrak{X}$, respectively. This inconsistency can confuse the token consistency check, leading to errors in the model's output.

9. In-Depth Methodology for Evaluating Tiny-Doc-Math

9.1. Construction of Tiny-Doc-Math Dataset

The evaluation dataset is constructed primarily from arXiv papers in the fields of mathematics and computer science, published after June 2024. We manually select a batch of these papers and download the LaTeX source code and corresponding PDFs. Using regular expressions, we match the formulas displayed from the LaTeX source. After individual formula rendering and manual verification, the Tiny-Doc-Math validation set is built, comprising 12 papers, 196 pages, and a total of 437 formulas.

9.2. Formula-Level Evaluation Methodology

Once the evaluation dataset is constructed, we extract mathematical formulas from the LaTeX source code. Since LaTeX sources may contain custom commands and comments from authors, we apply a series of preprocessing steps to ensure accurate extraction. First, we remove comments from the LaTeX source using regular expressions (including `"%"`, `"\iffalse... \fi"`, and `"\begin{comment}...\end{comment}"`). Next, we convert aliases defined by commands such as `"\newcommand{}{}"`, `"\renewcommand{}{}"`,

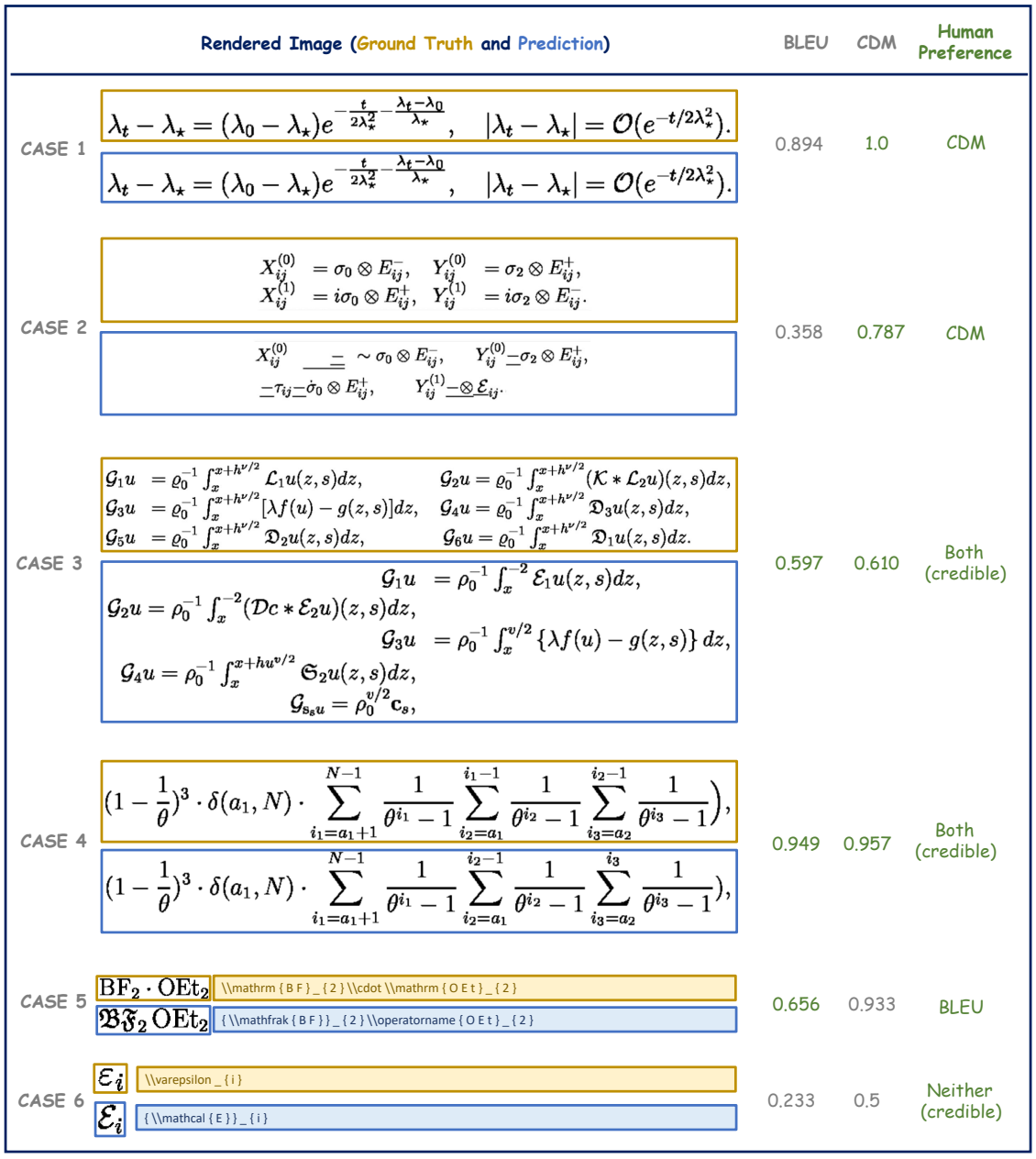

Figure 6. Examples of different human preferences (**CDM**, **BLEU**, **Both** (credible), **Neither** (credible)). Case 5 and Case 6 highlight some erroneous instances of CDM. In Case 5, CDM overlooks differences in character rendering styles, treating "`\mathfrac{BF}`" as identical to "`\mathrm{BF}`", despite their visual differences. Conversely, in Case 6, CDM distinguishes between "`\mathcal{E}`" and "`\varepsilon`", although they render similarly to human perception.

"`\DeclareMathOperator{}{}`", "`\DeclareMathOperator*{}{}`", "`\def\...{}`", and "`\DeclareRobustCommand{}{}`" to their original forms to ensure successful formula rendering. We then remove content before "`\begin{document}`" to avoid matching irrelevant information. After preprocessing, we extract displayed mathematical formulas from the LaTeX source using a series of regular expressions, as shown in Figure 7(a). For each paper, the matched mathematical formulas are written to a text file, one formula per line.

We render the extracted GT mathematical formulas to obtain formula-level GT images, which are then used as inputs for Mathpix, UniMerNet, pix2tex, and GPT-4o to generate corresponding predictions. Finally, we compute met-

rics such as BLEU and CDM after matching the predictions with the GTs.

9.3. Document-Level Evaluation Methodology

We convert PDF pages to images and use these images as inputs for Mathpix and GPT-4o to generate corresponding predictions, while Nougat takes the whole PDF as input. After obtaining the document-level predictions, we used extraction algorithms to extract displayed formulas from the predictions, and match them with the GT formulas obtained in the previous section to compute BLEU and CDM metrics.

Due to the different syntax formats of the outputs from different models, we use different regular expressions to extract formulas for each model, as shown in Figure 7(b), (c), and (d). Similarly, for each PDF, the matched mathematical formulas from each model's predictions are written to a text file, one formula per line.

9.4. Matching and Metric Computation

After obtaining the GTs and predicted mathematical formulas, we match the GTs with the predicted formulas line by line to compute the final CDM metric. Given the high accuracy of displayed formula predictions, we use edit distance as the metric for matching formulas. To account for different math delimiters used by different models (*e.g.*, `"\begin{equation}...\end{equation}"` vs. `"\[...\]"`), we remove all math delimiters before matching, focusing solely on the content. Labels and tags are also removed from the formulas.

The matching process consists of two rounds. In the first round, we set a low edit distance threshold for precise matching. This means that only predictions with a high similarity to the ground truth formula will be matched. We iterate through the GT formulas, calculating the edit distance with all predicted results. The prediction with a minimum edit distance is recorded as matched only if the minimum edit distance was below the threshold. If not, we skip the line and mark both the GT and the prediction as unmatched. In the second round, we set a higher threshold to account for those matching cases where the edit distance might be large. We iterate through the unmatched GT formulas, calculate the edit distance with the remaining unmatched predicted formulas, and record matches if the distance is below the threshold. If any predicted formulas remain unmatched after the first two rounds, we mark them as incorrect or redundant predictions and append them to the end of the matched results.

Through practical implementation, we find that setting the first-round threshold to 0.4 and the second-round threshold to 0.8 provides the most reasonable matching. Although extreme cases might occur where the rendered results are identical but fail to match due to large edit distances, these instances are not common and have been manually corrected.

After matching the GTs and predicted formulas, we compute metrics such as BLEU and CDM.

9.5. Result Discussion

As shown in Figure 8, GPT-4o's document-level predictions exhibited a significant number of CDM scores between 0.6 and 0.9, primarily due to hallucination phenomena in large models. For example, as shown in Figure 9(a), GPT-4o generates structurally similar but content-irrelevant results. Additionally, as shown in Figure 9(b), GPT-4o's predictions often lack standardized formatting, *i.e.*, frequently generating formulas without math delimiters, leading to extraction and rendering failures and resulting in many CDM=0 cases. For Mathpix, although the CDM between the document level and formula level is close, the proportion of CDM=1 predictions at the formula level is significantly lower. This is mainly due to the lack of commas in Mathpix's single formula predictions, as shown in Figure 9(c). Nougat's predictions often contain syntax errors, as shown in Figure 9(d), leading to rendering failures and CDM=0 cases. Moreover, Nougat's predictions sometimes leave several pages in the middle of the PDF with no prediction results, resulting in missing formulas in the final output.

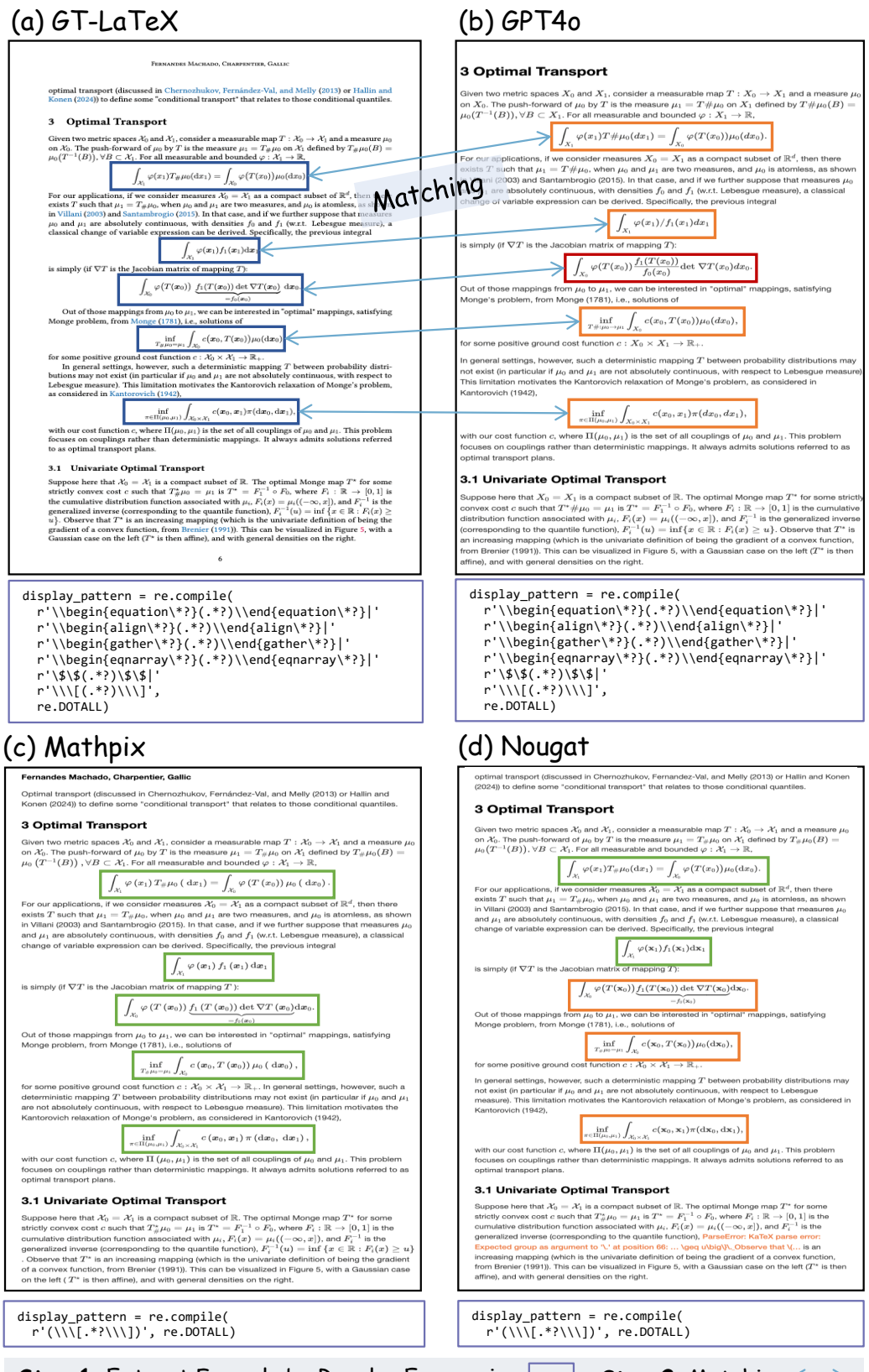

Figure 7. Detailed Process of Document-Level Evaluation.

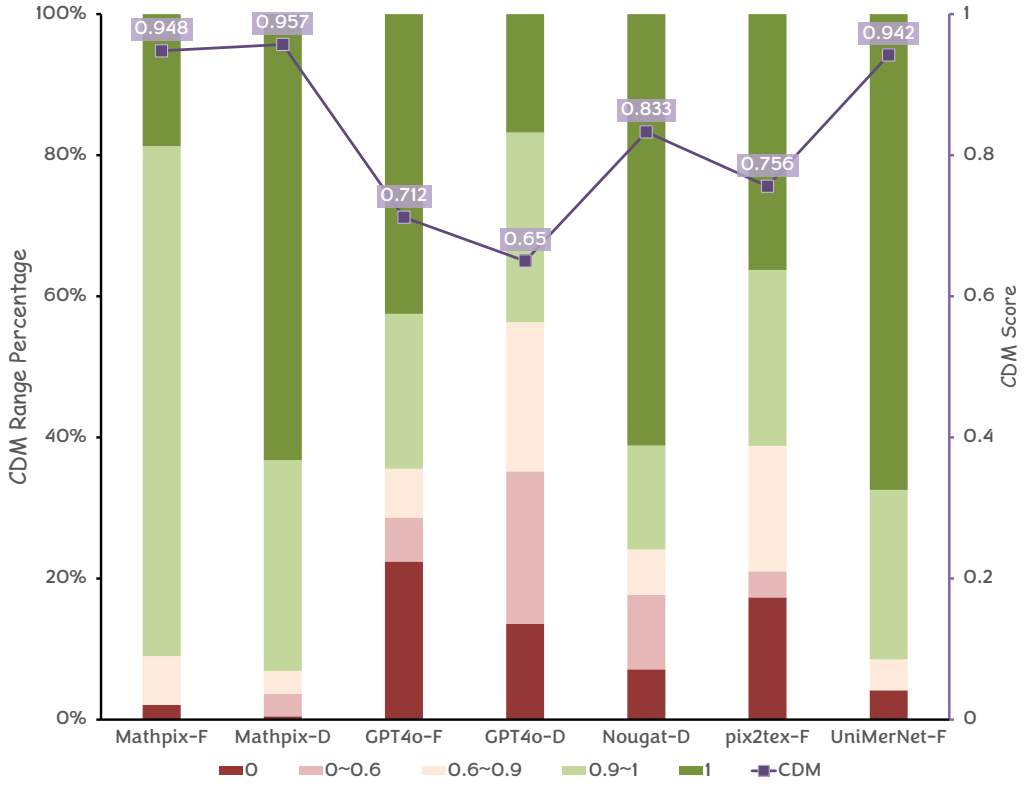

Figure 8. The CDM range percentage and CDM score of each model. "F" indicate Formula-Level, "D" indicate Document-Level.

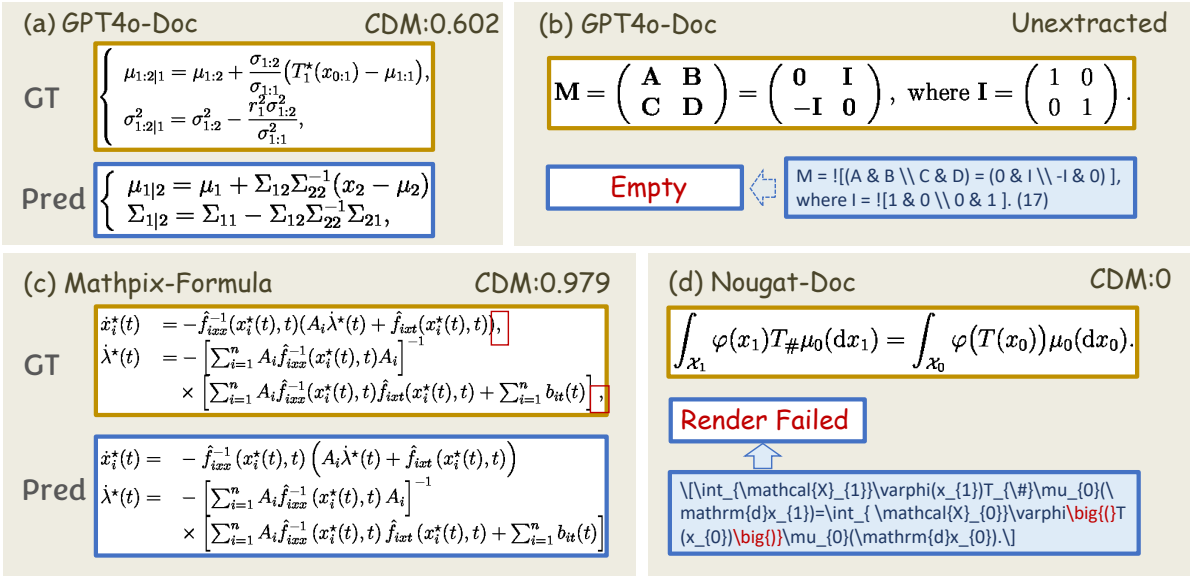

Figure 9. Examples of common prediction errors in GPT-4o, Mathpix, and Nougat.

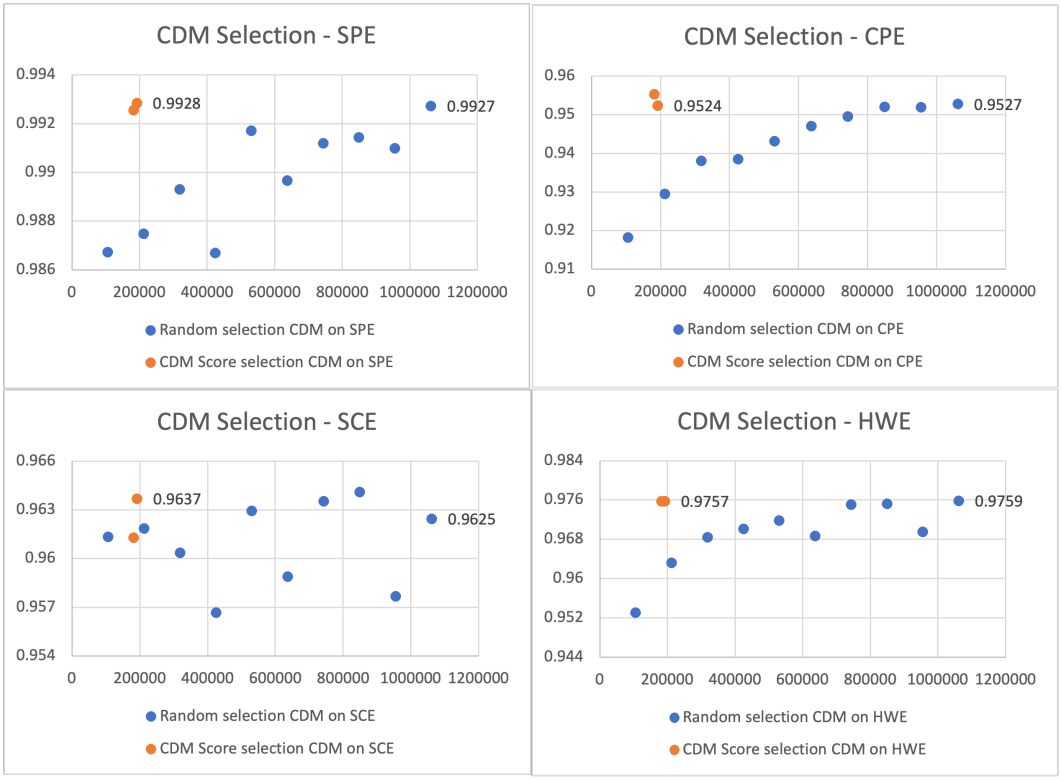

Figure 10. CDM metrics on four UniMER-Test subsets (SPE, CPE, SCE, HWE) for models trained with varying amounts of data (10%, 20%, ..., 100%) and models trained using two rounds of hard case selection. The scatter plot shows performance improvements with increasing training data and the efficiency of hard case selection.

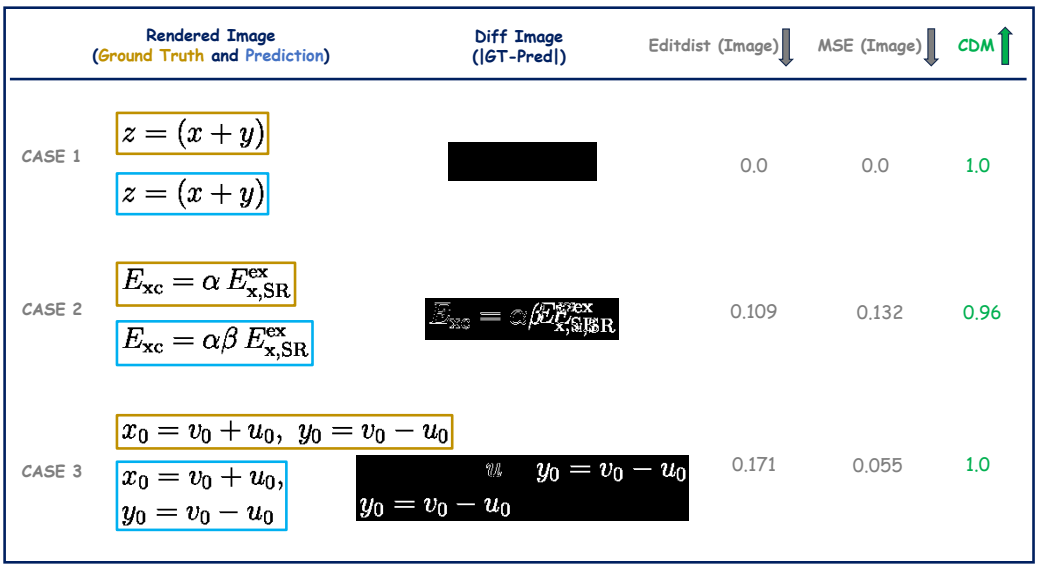

Figure 11. Examples of formula recognition evaluation using image edit distance and MSE. Case 1: Correct prediction with zero Editdist and MSE. Case 2: Missing one α causing all subsequent positions to mismatch. Case 3: Correct formula content but an extra newline character causing significant image difference.

10. Efficient Data Selection for Formula Recognition

Current formula recognition methods often overlook the importance of sample selection during training. We demonstrate that by utilizing the CDM metric for training data selection, it is possible to achieve performance comparable to using the entire dataset while only utilizing less than 20% of the data. We conduct the following experiment: First, we randomly split the UniMER-1M dataset into ten equal parts. We then train the model using 10%, 20%, up to 100% of the data and observe the model's performance with varying amounts of training data. As shown by the blue points in Figure 10, the model's performance generally improves as the amount of training data increases. Notably, with just 10% (106,179 samples) of the data, the model achieves satisfactory performance, accurately predicting most formulas. This suggests that the remaining 90% of the data may be largely redundant for training purposes.

To further investigate, we perform two rounds of hard case data selection. First, we use the model trained on 10% of the data to identify samples with CDM $\neq 1$ from the remaining 90%. We find 76,026 such samples, which is less than 8% of the remaining data, indicating that over 90% of the formulas can be accurately predicted. Combining these with the initial 10% random data, we have a total of 182,205 samples (17.16% of the UniMER-1M dataset). As shown in Figure 10, the model trained on this combined dataset performs comparably to the model trained on the full dataset, except for a slight underperformance on the SCE subset.

Next, we use this model to further select hard cases from the remaining data, identifying an additional 9,734 samples, representing about 1% of the remaining data. This brings the total to 191,939 samples (18.08% of the full dataset). The performance of this model shows a slight improvement over the previous round, achieving results comparable to or even exceeding those of the model trained on the full dataset across various subsets.

This experiment demonstrates the effectiveness of using CDM for hard case selection in formula recognition. Training based on hard case mining can serve as an efficient method to enhance model performance. This approach allows for the expansion of training data by selecting only the necessary samples, eliminating the need to use the entire dataset. Future formula recognition datasets can be expanded using this method, focusing on the most challenging samples to improve model accuracy and efficiency.

11. Evaluation Method Based on Image Differences

Previous work [32] mentions using image-based difference methods for evaluating formula recognition results, but a thorough analysis of the limitations of this approach is needed. To further assess the effectiveness of these methods, we conduct experiments using both image edit distance (Editdist) and Mean Squared Error (MSE) of image differences. As shown in Figure 11, Case 1 demonstrates that when the model's prediction is correct and the rendered output perfectly matches the ground truth (GT), both EditDist and MSE are zero, indicating an accurate formula. However, in Case 2, where the prediction misses the character α, the image-based difference method flags all subsequent positions as mismatched, even though only one character is missing. A more severe example is illustrated in Case 3, where the predicted formula content is correct but an extra newline character is predicted, leading to a significant image difference. In this case, both EditDist and MSE are non-zero and fail to reflect the error accurately. This highlights the necessity of the proposed CDM metric.

12. Latest UniMERNet performance

Table 2 shows how UniMERNet [30] compares to other models. It was recently updated with three model weights of different sizes, which we re-evaluated, and the results are shown in Table 4.

Method	SPE		CPE		HWE		SCE	
	CDM ↑	ExpRate@CDM ↑	CDM ↑	ExpRate@CDM ↑	CDM ↑	ExpRate@CDM ↑	CDM ↑	ExpRate@CDM ↑
Pix2tex	0.9619	0.7240	0.6489	0.0705	0.2453	0.0060	0.6762	0.3284
Texify	0.9852	0.9104	0.7041	0.2821	0.5269	0.2359	0.7932	0.5132
Mathpix	0.9729	0.4400	**0.9671**	0.288	0.9318	0.5928	0.9238	0.7233
UniMERNet-tiny	<u>0.9910</u>	0.9232	0.9491	0.6988	0.9328	0.6186	<u>0.9384</u>	0.7655
UniMERNet-small	0.9906	**0.9335**	0.9588	<u>0.7767</u>	<u>0.9370</u>	<u>0.6393</u>	**0.9406**	<u>0.7693</u>
UniMERNet-base	**0.9914**	<u>0.9329</u>	<u>0.9595</u>	**0.8046**	**0.9400**	**0.6431**	0.9373	**0.7697**

Table 4. Newest UniMER-Test subset evaluation results