DIFFMATH: SYMBOL- AND GRAPH-AWARE LATENT DIFFUSION TRANSFORMER FOR HANDWRITTEN MATHEMATICAL EXPRESSION GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Handwritten Mathematical Expression Generation (HMEG) remains challenging because it requires accurate modeling of complex hierarchical structures and two-dimensional layouts. Existing methods typically rely on explicit spatial supervision to model expression layouts, resulting in high annotation costs and limited scalability. To address this, we propose **DiffMath**, a structure-aware latent diffusion framework that leverages structural priors derived from LaTeX, thereby eliminating the need for positional supervision. First, we design a **Rel**ational **A**bstract **S**yntax **T**ree (**RelAST**), a generation-oriented structural representation that distills MathML trees into compact triplet sequences encoding symbol identities, spatial relations, and nesting depths. Building upon RelAST, we introduce **MathVAE**, which learns a structure-preserving latent space through symbol-aware and relation-aware perceptual regularization, jointly encoding symbol semantics and structural topology. Finally, **MathDiT** performs conditional denoising in the learned latent space conditioned on RelAST, further guided by a global symbol-count prior via Adaptive Layer Normalization (AdaLN) to improve structural coherence. Experiments demonstrate that DiffMath generates structurally consistent handwritten mathematical expressions, consistently outperforms existing HMEG methods, and effectively improves downstream OCR performance through synthetic data augmentation. The code will be publicly available.

1 INTRODUCTION

Mathematical expressions are ubiquitous in education, scientific research, and are commonly represented using LaTeX. Unlike plain text, mathematical expressions exhibit rich hierarchical structures and spatial relationships, which can be compactly encoded by LaTeX through specialized syntax. Handwritten Mathematical Expression Generation (HMEG) aims to synthesize realistic handwritten trajectories of mathematical expressions from LaTeX strings, as illustrated in Fig. 1(a). HMEG has broad applications in educational content creation and data augmentation for Handwritten Mathematical Expression Recognition (HMER). Compared with conventional augmentation methods Guan et al. (2025) that mainly perturb existing handwritten samples, HMEG can synthesize diverse handwriting for previously unseen mathematical expressions, thereby expanding both content and structural diversity and providing more robust gains for downstream recognition.

As a representative structured generation task, HMEG requires the generated expressions to be simultaneously content-correct, structurally accurate, and visually realistic. Unlike handwritten text generation, mathematical expressions exhibit complex two-dimensional layouts and long-range structural dependencies, where even slight structural errors may alter their semantic meaning. Some methods (Chen et al., 2024; Gan et al.,

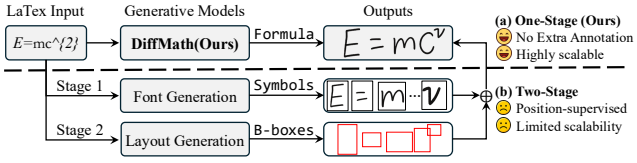


Figure 1: Unlike decoupled two-stage approaches that require explicit position-level supervision, **DiffMath (Ours)** adopts a streamlined end-to-end framework to directly map LaTeX to formula, reducing data dependency while improving global structural consistency.

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

Table 1: Comparison of different LaTeX processing strategies for generative modeling in terms of syntax support, position modeling, and generation quality. Our method explicitly models positional information while supporting arbitrary LaTeX syntax and achieving better generation quality. ExpRate denotes expression accuracy, while FID measures distribution distance.

Content Input	Position Modeling	Syntax Support		Generation Quality	
		Coverage	Extensible	ExpRate ↑	FID ↓
Raw LaTeX	None	Full	✓	56.27	9.62
Syntax Tree (Zhu et al., 2025)	Explicit	Limited	✗	–	–
MathML (Miller, 2026)	Implicit	Full	✓	62.27	7.57
RelAST (Ours)	Explicit	Full	✓	**70.70**	**5.43**

2025) therefore adopt a two-stage pipeline (Fig. 1(b)) that separately generates symbols and predicts their spatial layouts. However, such decoupled modeling requires fine-grained layout annotations and often produces inconsistencies between symbol content and structural composition.

To overcome these limitations, we propose **DiffMath**, an end-to-end latent diffusion framework that directly learns HMEG from paired LaTeX and trajectory data without requiring additional layout annotations. Our key insight is that the performance of end-to-end HMEG critically depends on the structural representation used as the generation prior. As shown in Tab.1, existing representations are not well suited for generation: raw LaTeX treats mathematical expressions as linear text, syntax-tree representations(Zhang et al., 2020) have limited syntax coverage, and standard MathML introduces redundant structural nodes that reduce information density. Motivated by these observations, we introduce Relational Abstract Syntax Tree (**RelAST**), a generation-oriented structural representation that explicitly encodes symbol identities, spatial relations, and structural hierarchy, providing compact and informative conditions for HMEG.

Building upon RelAST, DiffMath first employs **MathVAE**, a Variational Autoencoder that learns a structure-aware latent space for handwritten mathematical expressions. Inspired by DiffInk (Pan et al., 2026), MathVAE extends reconstruction-based latent learning with symbol-aware and relation-aware regularization, encouraging the latent space to jointly preserve handwritten appearance, symbol semantics, and structural dependencies. The resulting structure-aware latent space provides an effective foundation for subsequent diffusion-based generation.

Building upon the learned structure-aware latent space, DiffMath further employs **MathDiT**, a conditional latent diffusion Transformer for HMEG. MathDiT performs denoising in the latent space while being conditioned on the explicit symbol, relation, and depth representations provided by RelAST. To further improve global structural consistency, we incorporate a symbol-count prior (Li et al., 2022), which is jointly embedded with the diffusion timestep through Adaptive Layer Normalization (AdaLN) (Peebles & Xie, 2023). By combining local structural conditions with global expression-level priors, MathDiT generates handwritten mathematical expressions with improved structural accuracy and visual fidelity.

Our contributions are summarized as follows: 1) We propose **DiffMath**, an end-to-end latent diffusion framework for HMEG that leverages structural priors derived from LaTeX, eliminating the need for dense positional supervision. 2) We design **RelAST**, a generation-oriented structural representation that transforms MathML trees into compact symbol–relation–depth triplets, providing expressive and efficient structural conditions for HMEG. 3) We introduce a structure-aware latent diffusion framework that learns structure-preserving latent representations through symbol-aware and relation-aware regularization, and further enhances generation with both local structural conditions and a global symbol-count prior. 4) Extensive experiments demonstrate that DiffMath achieves state-of-the-art performance on the MathWriting benchmark and that the generated samples consistently improve downstream HMER through synthetic data augmentation.

2 Related Works

Handwritten Text Generation has progressed from RNN-based sequential modeling (Graves, 2013) to Transformer- and diffusion-based frameworks (Pippi et al., 2023; Dai et al., 2023; Bhunia et al., 2021; Pippi et al., 2025; Nikolaidou et al., 2025; Dai et al., 2025a;b). Recent methods (Ren et al., 2025; Pan et al., 2026) further improve generation quality through explicit layout modeling or

structured latent diffusion. However, these approaches are designed for 1D linear text and are not well suited to mathematical expressions, which require modeling complex 2D topological structures.

Handwritten Mathematical Expression Generation. Early methods (Springstein et al., 2021) formulated HMEG as image translation, while subsequent approaches (Chen et al., 2024; Gan et al., 2025; Wang et al., 2026) improved structural accuracy through explicit spatial modeling or style transfer. However, these methods rely on costly dense supervision, such as character-level bounding boxes or style labels. In contrast, DiffMath exploits structural priors derived from LaTeX parsing to model complex 2D topologies without positional annotations.

Handwritten Mathematical Expression Recognition. Most HMER systems adopt encoder-decoder frameworks for handwritten mathematical expression recognition. Representative methods include BTTR (Zhao et al., 2021), CoMER (Zhao & Gao, 2022), CAN (Li et al., 2022), SAN (Yuan et al., 2022), TAMER (Zhu et al., 2025), PosFormer (Guan et al., 2025), and Uni-MuMER (Li et al., 2025), which progressively incorporate counting, syntax-aware decoding, hierarchical modeling, and vision-language learning. However, these tree representations retain redundant non-terminal container nodes that are unnecessary for generation.

Text to Image Generation. Diffusion Transformers (DiTs)(Peebles & Xie, 2023) have become the dominant paradigm for high-fidelity generation and underpin many recent text-to-image systems(Podell et al., 2024; Wu et al., 2025; Team, 2025b; Labs et al., 2025; Labs, 2025). Inspired by their success, we adopt a latent diffusion Transformer for HMEG, conditioned on a structured textual representation to better preserve symbolic correspondence and hierarchical mathematical layouts.

3 METHOD

3.1 PRELIMINARY

Data Format. A handwritten mathematical expression consists of a LaTeX expression and a trajectory sequence $X \in \mathbb{R}^{N\times 4}$. The LaTeX expression is a linear string encoding the symbolic structure of the formula. Each trajectory point is represented as (x, y, pen), where (x, y) denotes the spatial coordinates and pen is a one-hot vector indicating the pen state (*Pen Down* or *Pen Up*).

HMEG Task Overview. We formalize the task as follows: given a source LaTeX sequence, the goal is to generate an online handwriting trajectory X that accurately renders the formula. The objective is to produce a sequence of points that not only captures the semantic essence of the mathematical expression but also adheres to the complex spatial-structural constraints inherent in its LaTeX representation.

3.2 TOWARD A GENERATION-ORIENTED REPRESENTATION

Motivation. LaTeX linearizes mathematical logic into a structured 1D sequence, but often hides the 2D spatial relations among symbols, making it less effective when directly used as a generation condition. As summarized in Tab. 1, raw LaTeX encodes layouts implicitly and leads to suboptimal generation quality. In HMER (Li et al., 2025; Zhu et al., 2025; Yuan et al., 2022; Zhang et al., 2020), syntax-tree parsing improves structural accuracy by explicitly modeling expression layouts. This observation suggests that structural parsing can also serve as a useful prior for generative modeling, where faithful layout synthesis is crucial. However, existing LaTeX processing strategies are not directly suited for generation: rule-based syntax trees have limited coverage, and MathML trees introduce redundant hierarchy without directly exposing spatial relations. These limitations motivate us to design RelAST, a compact generation-oriented representation that retains broad syntax coverage while explicitly encoding symbol identities, 2D relations, and hierarchical layouts.

LaTeX to RelAST. As illustrated in Fig. 2, we first normalize LaTeX into a standardized MathML representation using LaTeXML (Miller, 2026) (**Step 1**), avoiding the limited coverage and inconsistent formats of rule-based parsers (Li et al., 2025; Zhu et al., 2025; Yuan et al., 2022; Zhang et al., 2020). We then remove non-terminal container nodes and perform depth-first pre-order traversal to obtain relational triplets $[S, R, D]$ (**Step 2**), where S, R, and D denote the visible symbol, spatial relation, and hierarchical depth, respectively. The resulting representation explicitly preserves both 2D layouts and hierarchical structures, enabling the model to distinguish ambiguous expressions such as a^{bc} and $a^b c$. The first relation is initialized as R_{root}. Representative examples of symbol and relation tokens are listed in Tab. 3.

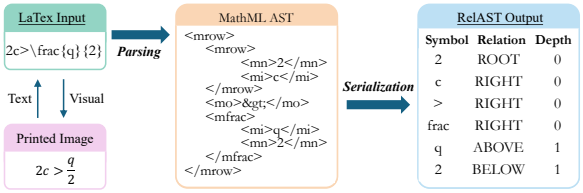


Figure 2: **RelAST construction.** LaTeX is converted into a standardized MathML tree and serialized via depth-first traversal into triplets $[S, R, D]$, representing symbol content (S), spatial relations (R), and hierarchical depth (D).

Figure 3: Examples of different token types used in RelAST.

Token Type	Examples
Structural relations	`<RIGHT>`, `<SUP>`, `<SUB>`, `<ABOVE>`, `<BELOW>`, `<INSIDE>`
Layout tokens	`frac`, `sqrt`, `root`, `table`, `<ROW>`, `<CELL>`
Numbers	$0, 1, 2, 3, 4, 5, \ldots$
Latin variables	$x, y, z, a, b, c, A, B, C, \ldots$
Greek symbols	$\alpha, \beta, \gamma, \theta, \lambda, \pi, \ldots$
Arithmetic operators	$+, -, =, \times, \div, \cdot, \ldots$
Calculus operators	$\int, \sum, \prod, \partial, \ldots$
Delimiters	$(), [], \{\}, \vert\vert, \ldots$

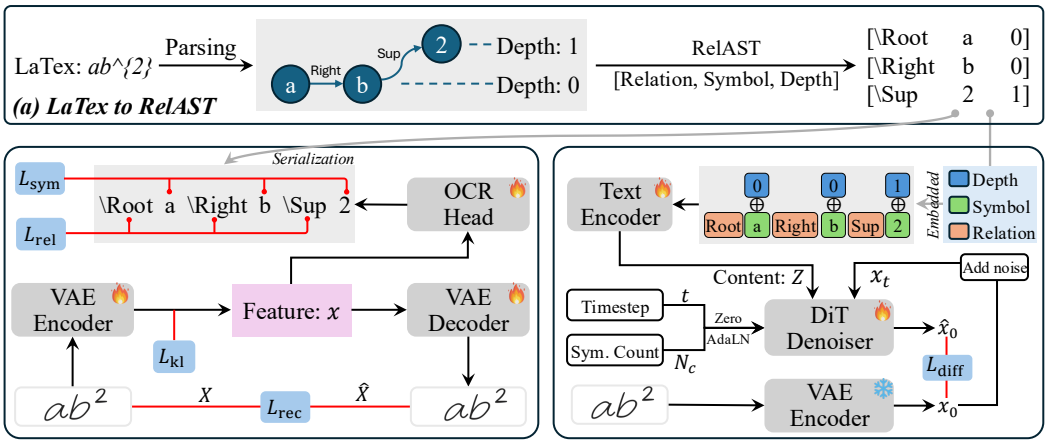


Figure 4: **Overview of the DiffMath Framework.** (a) LaTeX is first parsed into an explicit structured representation, termed RelAST. (b) MathVAE compresses raw trajectories into a latent space- and introduces symbol- and relation-level perceptual losses ($\mathcal{L}_{\text{sym}}, \mathcal{L}_{\text{rel}}$) to enrich its semantic and structural representations. (c) MathDiT reconstructs the clean latent $\hat{x}_0$ from noise, conditioned on structural tokens and global counts for high-fidelity formula generation.

3.3 Overview of the DiffMath

As shown in Fig. 4(a), input LaTeX is first parsed into RelAST, which encodes symbols, spatial relations, and nesting depths as structural guidance. Based on this representation, DiffMath consists of two main components: (1) MathVAE (Fig. 4(b)), a pre-trained sequential VAE for learning a structure-aware latent space; and (2) MathDiT (Fig. 4(c)), a conditional latent diffusion Transformer for structure-guided denoising. MathVAE compresses raw online trajectories $X \in \mathbb{R}^{N\times 4}$ into latent representations $x \in \mathbb{R}^{l\times d}$ that preserve both symbol-level geometry and inter-symbol topology. In this latent space, MathDiT performs conditional denoising from x_t, which is obtained by perturbing x during training or sampled from Gaussian noise during inference. Conditioned on the structural tokens Z and the global symbol count N_c, MathDiT predicts the clean latent representation $\hat{x}_0$, which is decoded by MathVAE into the final handwritten mathematical expression.

3.4 Symbol- and Graph-Aware VAE.

As shown in Fig. 4(b), MathVAE adopts an encoder–decoder architecture to map raw handwriting trajectories into a compact latent space. Rather than learning a purely reconstruction-oriented representation, MathVAE is designed to produce semantically informative latent features that facilitate downstream conditional diffusion. Motivated by recent studies on representation learning for generation and transfer (Caron et al., 2021; Siméoni et al., 2026; Yao et al., 2025; Pan et al., 2026; Sclocchi et al., 2025), we incorporate symbol- and relation-aware regularization, enabling the latent space to jointly encode handwritten appearance, symbol semantics, and spatial structure.

Sequence Encoder. To extract informative features from complex handwritten mathematical expressions, we employ a 1D convolutional encoder to transform the input handwriting trajectory sequence $X \in \mathbb{R}^{N\times 4}$ into a compact latent representation $x \in \mathbb{R}^{l\times d}$. This representation is then fed into two complementary branches: (1) a trajectory decoder for reconstructing low-level handwriting trajectories at the point level, and (2) a Transformer-based HMER module for capturing high-level semantic information, including symbolic content and structural relationships.

Sequence Decoder. The decoder first applies a 1D convolutional stack that mirrors the encoder, up-sampling the latent feature x into a feature map $O_t \in \mathbb{R}^{N\times(6p+2)}$. At each timestep, O_t is divided into two branches: (1) $6p$ parameters for a p-component Gaussian Mixture Model (GMM) that predicts the coordinate distribution of (x, y), and (2) 2 logits for classifying the pen state, i.e., *Pen Down* or *Pen Up*.

Reconstruction Objective. The reconstruction objective consists of a coordinate loss $\mathcal{L}_{\text{gmm}}$ and a pen-state loss $\mathcal{L}_{\text{pen}}$. Given the ground-truth point $\mathbf{p}_t = (x_t, y_t)$, the coordinate loss is defined as the negative log-likelihood under the predicted GMM. As shown in equation 1, where Ω denotes the set of valid non-padded trajectory steps, for each mixture component m, the GMM branch predicts the mixing coefficient $\pi_{m,t}$, mean $\boldsymbol{\mu}_{m,t}$, and covariance $\boldsymbol{\Sigma}_{m,t}$.

$$\mathcal{L}_{\text{gmm}} = -\frac{1}{|\Omega|}\sum_{t\in\Omega}\log\left(\sum_{m=1}^{p}\pi_{m,t}\mathcal{N}(\mathbf{p}_t \mid \boldsymbol{\mu}_{m,t}, \boldsymbol{\Sigma}_{m,t})\right), \tag{1}$$

The pen-state branch is optimized with a focal loss (Lin et al., 2017) defined as equation 2, where $y^{\text{pen}}_{t,k}$ and $p^{\text{pen}}_{t,k}$ denote the ground-truth label and predicted probability of pen-state class k, respectively.

$$\mathcal{L}_{\text{pen}} = -\frac{1}{N}\sum_{t=1}^{N}\sum_{k=1}^{2}\alpha_k(1-p^{\text{pen}}_{t,k})^{\gamma}y^{\text{pen}}_{t,k}\log p^{\text{pen}}_{t,k}, \tag{2}$$

The final reconstruction loss is defined as equation 3, We compute $\mathcal{L}_{\text{gmm}}$ over valid trajectory steps, while $\mathcal{L}_{\text{pen}}$ is applied across the entire sequence, including padded regions, enabling the decoder to learn pen transitions and stopping behavior.

$$\mathcal{L}_{\text{rec}} = \mathcal{L}_{\text{gmm}} + \mathcal{L}_{\text{pen}}. \tag{3}$$

Symbol- and Graph-Aware Regularization. To further align the latent space with mathematical structures, we introduce a task-aware perceptual loss $\mathcal{L}_{\text{per}}$ as a structural regularizer. This loss consists of two complementary components: a symbol-level perceptual loss and a structure-level perceptual loss. Although they focus on different aspects of handwritten mathematical expressions, symbols and structures are inherently coupled: structural relations are defined over symbols, and mathematical structures become meaningful only when instantiated with concrete symbolic content. Therefore, we integrate symbol perception and relation perception into a unified OCR-based recognition framework, encouraging the encoder to learn representations that are both semantically discriminative and topology-aware.

Specifically, we employ a Transformer-based OCR module to predict an interleaved structural sequence $[R_{\text{root}}, S_1, R_2, S_2, \ldots, R_n, S_n]$, which is obtained by linearizing the RelAST representation. Here, S_i denotes the i-th symbol token, and R_i represents its associated spatial relation. The predicted probability sequence $\mathbf{P}^{\text{ocr}}$ is optimized against the ground-truth structural sequence $\mathbf{y}^{\text{ocr}}$ using Connectionist Temporal Classification (CTC (Graves et al., 2006)) loss. This objective jointly enforces symbol-level recognition and relation-level topology consistency:

$$\mathcal{L}_{\text{per}} = \mathcal{L}_{\text{sym/rel}} = \text{CTC}\left(\mathbf{P}^{\text{ocr}}, \mathbf{y}^{\text{ocr}}\right). \tag{4}$$

By optimizing $\mathcal{L}_{\text{sym}}$ and $\mathcal{L}_{\text{rel}}$ within a single sequence prediction objective, the encoder is encouraged to perceive a formula as a cohesive structural entity, leading to latent representations that better preserve both symbol identities and spatial layouts.

VAE Loss Function. MathVAE is trained end-to-end with the objective function $\mathcal{L}_{\text{vae}}$ defined in Eq. 5. Besides reconstruction and perceptual regularization, we include a KL divergence term following the variational autoencoder formulation (Kingma & Welling, 2013), which regularizes the

latent distribution toward a standard Gaussian prior. This enables the encoder to learn structurally coherent and semantically aligned latent representations for diffusion-based handwriting synthesis.

$$\mathcal{L}_{\text{vae}} = \sum \lambda_\ell \cdot \mathcal{L}_\ell \quad \text{where } \ell \in \{\text{rec, kl, per}\} \tag{5}$$

3.5 HMEG VIA MATHDIT

As shown in Fig. 4 (c), MathDiT employs a Transformer-based diffusion architecture for HMEG within the latent space. Built upon the compact and structured manifold of MathVAE, MathDiT progressively refines noisy latent variables into coherent stroke trajectories through an iterative denoising process. Unlike traditional models conditioned on raw LaTeX strings, MathDiT is conditioned on the proposed RelAST, which provides an explicit structural prior through its serialized triplets $[S, R, D]$. By leveraging this topology-aware input, MathDiT achieves precise and structural generation, ensuring high fidelity and superior readability even for complex, deeply nested mathematical expressions.

Structural Text Representation. Compared with raw LaTeX input, we explicitly model symbols and relational nodes to capture spatial structures. By parsing each LaTeX expression into RelAST, we obtain its triplet representation $[S, R, D]$, which serves as the structured textual sequence. Specifically, symbols S and relations R are projected into high-dimensional latent spaces via separate embedding layers. To capture the recursive hierarchy of mathematical notation, the nesting depth D is encoded as a depth embedding. This depth information is integrated with its corresponding symbol embedding S through element-wise addition, whereas relation nodes remain depth-invariant. The resulting sequence follows an interleaved structural pattern: $[R_{\text{root}}, S_1 + D_1, R_2, S_2 + D_2, \dots]$, originating from a dedicated R_{root} node. This selective integration of depth-aware embeddings allows the model to anchor symbols within their respective hierarchical levels while using relations as topological bridges, strengthening its capacity for modeling complex, multi-scale mathematical structures.

Complexity-Sensitive Symbol Count Prior. Long mathematical expressions are more challenging to generate due to their increased structural complexity. We therefore use the number of entity symbols in RelAST as a global complexity prior and incorporate it into the diffusion process through a *Complexity-Sensitive Time Modulation* mechanism.

Let N_c denote the number of entity symbols derived from RelAST. Instead of directly concatenating this signal with the structural condition, we adaptively rescale the timestep embedding according to the symbol count:

$$\hat{\mathbf{t}} = \text{MLP}(\mathbf{t}) \odot (1 + \tanh(\alpha)\,(1 - t/T)\log(1 + N_c))\,, \tag{6}$$

where α is a learnable scalar, t is the current diffusion timestep, and T is the total number of diffusion steps. The logarithmic mapping stabilizes the symbol-count signal, while $\left(1 - \frac{t}{T}\right)$ gradually increases its influence during denoising. As a result, the model first recovers the global layout and progressively incorporates expression complexity during later refinement, reducing symbol omission and over-generation in complex formulas.

Diffusion Modeling. MathDiT performs conditional denoising in the structured latent space learned by MathVAE. During training, a clean latent representation x_0 is first perturbed into a noisy latent x_t according to the diffusion schedule:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{7}$$

To incorporate structural guidance, the noisy latent x_t is concatenated with the RelAST condition Z along the channel dimension, and then projected to the hidden dimension of the Transformer. Given the timestep t and the symbol count N_c, MathDiT directly predicts the clean latent representation:

$$\hat{x}_0 = \text{MathDiT}_\theta([x_t, Z], t, N_c). \tag{8}$$

The model is optimized by minimizing the reconstruction error between the predicted clean latent $\hat{x}_0$ and the ground-truth clean latent x_0:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{x_0, t, \epsilon}\left[\|\hat{x}_0 - x_0\|_2^2\right]. \tag{9}$$

Table 2: **Quantitative comparison with baseline methods on the MathWriting benchmark.**

Methods	Venue	Edit.% ↑	Exp.% ↑	BLEU ↑	FID ↓	User% ↑
FormulaGAN (Springstein et al., 2021)	ICMR'21	95.06	65.86	92.14	98.15	2.6
One-DM (Dai et al., 2025a)	ECCV'24	88.20	39.66	82.02	29.33	4.1
SD-XL (Podell et al., 2024)	ICLR'24	88.93	35.00	81.39	9.50	7.9
FLUX.1 (Labs et al., 2025)	arXiv'25	91.06	53.12	85.21	16.06	10.4
Z-Image (Team, 2025b)	arXiv'25	78.82	22.21	61.51	8.04	5.3
Qwen-Image (Wu et al., 2025)	arXiv'25	94.63	65.37	91.83	10.32	21.4
DiffInk (Pan et al., 2026)	ICLR'26	92.33	58.35	90.15	8.24	17.0
DiffMath	**This Work**	**95.47**	**70.70**	**93.05**	**5.43**	**31.3**

Within the denoising blocks, the complexity-aware embedding $\hat{t}$ is injected via Adaptive Layer Normalization (AdaLN) (Peebles & Xie, 2023), enabling the refinement process to be dynamically modulated by global symbol density. Together with the RelAST condition Z, this design guides the denoising trajectory to satisfy both structural constraints and global complexity priors, yielding high-fidelity and topologically consistent expressions.

During inference, generation starts from Gaussian noise $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. MathDiT iteratively predicts the clean latent representation conditioned on Z and N_c, and progressively refines the latent through the reverse diffusion process. The final denoised latent is decoded by MathVAE into an online handwritten mathematical expression.

4 Experiments

4.1 Experimental Setup

Data Preparation. We utilize the recent and relatively challenging MathWriting (Gervais et al., 2025) dataset for HMEG. Our experimental setup uses 220k authentic human-written samples for training and a dedicated test set of over 7k samples. By parsing LaTeX into RelAST, we obtain a vocabulary of 244 tokens, including digits, Latin/Greek characters, operators, and relational symbols.

Baseline Methods. We select several domain-relevant generative methods, including FormulaGAN (Springstein et al., 2021), a GAN-based approach, as well as One-DM (Dai et al., 2025a) and DiffInk (Pan et al., 2026), which serve as strong baselines for handwritten content generation. We also include high-capacity text-to-image models, including SD-XL (Podell et al., 2024), FLUX.1 (Labs et al., 2025), Z-Image (Team, 2025b), and Qwen-Image (Wu et al., 2025), to assess generation quality.

Evaluation. HMEG requires both content fidelity and handwriting realism. We evaluate content fidelity using the state-of-the-art (SOTA) HMER model Uni-MuMER (Li et al., 2025) with Edit, Expression Recognition Rate (ExpRate), and Bilingual Evaluation Understudy (BLEU). Handwriting realism is assessed by Fréchet Inception Distance (FID) (Heusel et al., 2017) and a user study on content correctness, structural correctness, and handwriting style.

Implementation Details. MathVAE is trained using the AdamW optimizer with an initial learning rate of 1.5×10^{-4}, while MathDiT uses AdamW with an initial learning rate of 3×10^{-4}. The diffusion model employs $T = 1000$ timesteps with a cosine noise schedule. During training, classifier-free guidance (CFG (Ho & Salimans, 2021)) is enabled by randomly dropping the conditioning input with a probability of 10%. During inference, the guidance scale is set to 3.0, and DDIM (Song et al., 2021) sampling with 20 steps is adopted for efficient generation.

4.2 Main Results

Quantitative Evaluation. As shown in Tab. 2, for content fidelity, DiffMath (Ours) obtains an ExpRate of 70.70%, outperforming FormulaGAN by 4.84% and exceeding SD-XL (35.00%) and FLUX.1 (53.12%). These results suggest that the proposed RelAST representation provides more effective guidance than raw LaTex input. DiffMath also achieves the lowest FID score (5.43), indicating that the generated samples are closer to the real data distribution. Although large-scale models such as Qwen-Image produce visually plausible results, their structural accuracy remains lower.

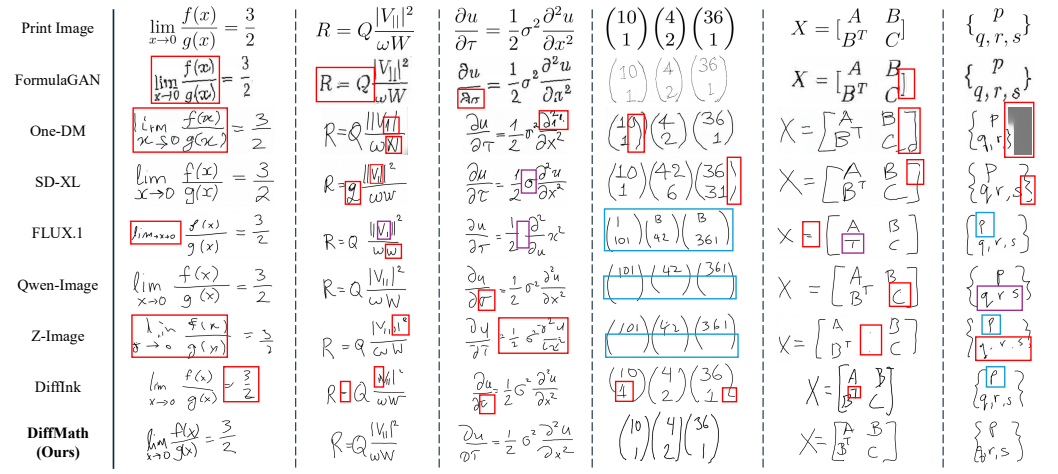


Figure 5: **Qualitative comparison results.** Red boxes highlight atomic content errors (incorrect symbols or artifacts), blue boxes indicate structural misalignments, and purple boxes denote missing symbols. Please zoom in for a better view of the details.

Table 3: **Quantitative ablation studies of DiffMath.** (a) Effect of MathVAE on latent representation learning. (b) Effect of structural priors in MathDiT.

(a) Effect of MathVAE

VAE Loss			Generation Performance			
$\mathcal{L}_{\text{rec+kl}}$	$\mathcal{L}_{\text{sym}}$	$\mathcal{L}_{\text{rel}}$	Edit	Exp	BLEU	FID
✓			89.77	46.23	81.55	18.24
✓	✓		92.75	65.22	91.55	6.56
✓	✓	✓	**95.47**	**70.70**	**93.05**	**5.43**

(b) Effect of MathDiT

Condition Inputs				Generation Performance			
Sym.	Rel.	Dep.	Glo.	Edit	Exp	BLEU	FID
✓				91.25	56.27	88.92	8.62
✓	✓			93.91	63.89	89.20	5.99
✓	✓	✓		95.46	69.83	92.65	5.44
✓	✓	✓	✓	**95.47**	**70.70**	**93.05**	**5.43**

Compared with DiffInk, a method for unstructured handwritten text-line generation, DiffMath improves ExpRate by 12.35%, highlighting the advantage of explicit structural modeling. This trend is further supported by the user study, where DiffMath receives the highest preference score.

Qualitative evaluation. As shown in Fig. 5, red and blue boxes highlight content and structural errors, respectively. While FormulaGAN struggle with realistic handwriting styles, more recent models like One-DM often suffer from background artifacts and character distortions. Even powerful pre-trained models (e.g., FLUX.1, Qwen-Image) frequently fail in dense symbol scenarios like matrices. Similarly, DiffInk exhibits structural collapse due to a lack of explicit layout modeling. In contrast, DiffMath maintains high-fidelity generation and robust structural integrity. Additional comparison results and generated samples can be found in Fig. 8 and Fig. 9.

4.3 Ablation Study of DiffMath

Quantitative analysis. Tab. 3 reports the quantitative ablation results of DiffMath. For MathVAE, introducing the symbol-aware regularization $\mathcal{L}_{\text{sym}}$ substantially improves the learned latent representation, increasing ExpRate by 19 points while reducing FID. Further incorporating the relation-aware regularization $\mathcal{L}_{\text{rel}}$ yields additional gains, achieving an ExpRate of 70.70% and an FID of 5.43. For MathDiT, progressively incorporating relation tokens **Rel.**, depth embeddings **Dep.**, and the global symbol-count prior **Glo.** consistently improves structural accuracy and generation quality, with the full model achieving the best performance across all evaluation metrics.

Visual analysis. Fig. 6 provides qualitative comparisons of different DiffMath variants. The left figure shows that the proposed MathVAE improves symbol correctness and structural consistency by learning a more informative latent representation. The right figure demonstrates that progressively introducing structural priors into MathDiT effectively reduces content omissions and layout errors, resulting in more accurate and coherent handwritten mathematical expressions.

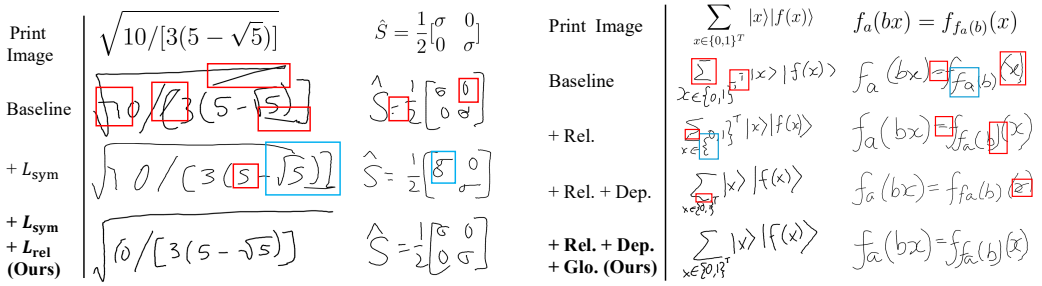


Figure 6: **Visual ablation studies.** The left and right figures compare different MathVAE and MathDiT variants, respectively. Red and blue boxes highlight content and structural/style errors. Incorporating all proposed components yields the best visual quality and structural consistency.

4.4 Data Augmentation for HMER

HMEG provides an effective way to augment training data for HMER while reducing annotation costs. To evaluate DiffMath as a data generator, we compare its synthetic data with conventional augmentation on MathWriting (Gervais et al., 2025) and the CROHME (Mouchère et al., 2014; 2016; Mahdavi et al., 2019) benchmark series, using sequence-level ExpRate together with CDM (Wang et al., 2025) and ExpRate@CDM. As shown in Tab. 4, we adopt a strong augmentation recipe from prior HMER studies (Guan et al., 2025), yet it provides only marginal gains, whereas DiffMath-generated samples consistently yield larger improvements across Qwen3-VL-8B-Instruct (Team, 2025a) and the SOTA Uni-MuMER (Li et al., 2025). This difference stems from the fact that conventional augmentation mainly perturbs existing samples and thus preserves the underlying expression distribution, while generative augmentation can leverage LaTeX expressions beyond the original training set to introduce new symbol combinations and spatial structures. This capability effectively expands both the quantity and structural diversity of training data, which is particularly valuable under the current data-scaling paradigm for improving recognition performance.

Table 4: **Downstream OCR experiments.** "Aug." denotes conventional data augmentation, while "Syn." denotes synthetic data generated by DiffMath (**Ours**). For "Syn.", we add 150k and 30k generated samples to the MathWriting and CROHME training sets, respectively.

OCR Model	Training Data	MathWriting		Avg. CROHME (14/16/19)	
		Exp.%	@CDM	Exp.%	@CDM
Qwen3-VL 8B-Instruct	Original Data	67.22	70.93	69.19	72.34
	+ Aug. Data	67.84$^{+0.62}$	71.05$^{+0.12}$	69.56$^{+0.37}$	72.37$^{+0.03}$
	+ Syn. Data	**68.58**$^{+1.36}$	**71.79**$^{+0.86}$	**72.40**$^{+3.21}$	**75.08**$^{+2.74}$
Uni-MuMER (Current SOTA)	Original Data	68.58	74.11	77.70	80.88
	+ Aug. Data	70.01$^{+1.43}$	74.38$^{+0.27}$	78.05$^{+0.35}$	80.95$^{+0.07}$
	+ Syn. Data	**71.42**$^{+2.84}$	**75.01**$^{+0.90}$	**79.17**$^{+1.47}$	**81.82**$^{+0.94}$

5 Conclusion

In this paper, we presented DiffMath, a structure-aware latent diffusion framework for HMEG that derives structural priors from LaTeX without requiring layout annotations or symbol-level positional supervision. By introducing RelAST, DiffMath explicitly models symbol identities, two-dimensional spatial relations, and hierarchical layouts, providing informative structural conditions for handwritten mathematical expression generation. Built upon this representation, MathVAE learns a structure-preserving latent space, while MathDiT performs structure-conditioned latent diffusion to generate handwritten mathematical expressions with high visual fidelity and structural consistency. Extensive experiments demonstrated that DiffMath consistently outperformed existing methods, particularly in preserving inter-symbol relations, superscript and subscript structures, and global expression layouts. Furthermore, the generated samples served as effective synthetic data for HMER, consistently improving multiple recognition models across different benchmarks. Notably, these gains were larger than those obtained with conventional data augmentation, suggesting the potential of generative data augmentation for enriching HMER training data. These results provide a perspective on structured generative modeling by highlighting the value of explicit structural priors.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing, as well as environment configuration and data processing during the reproduction of baseline methods. All AI-assisted outputs were reviewed and verified by the authors. We take full responsibility for the final content of this work.

ETHICS STATEMENT

All datasets used in this work are publicly available research datasets and are used in accordance with their original research purposes and licenses. We do not collect or release any additional personally identifiable information. Our method is developed for research on online handwriting generation, and generated samples are intended solely for legitimate research and educational use.

REPRODUCIBILITY STATEMENT

The supplementary material provides detailed descriptions of the model architecture and training configurations. In addition, we will publicly release the source code and model configurations to facilitate future research in this field.

REFERENCES

Ankan Kumar Bhunia, Salman Khan, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, and Mubarak Shah. Handwriting transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1086–1094, October 2021.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Yu Chen, Fei Gao, Yanguang Zhang, Maoying Qiao, and Nannan Wang. Generating handwritten mathematical expressions from symbol graphs: An end-to-end pipeline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15675–15685, June 2024.

Gang Dai, Yifan Zhang, Qingfeng Wang, Qing Du, Zhuliang Yu, Zhuoman Liu, and Shuangping Huang. Disentangling writer and character styles for handwriting generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5977–5986, June 2023.

Gang Dai, Yifan Zhang, Quhui Ke, Qiangya Guo, and Shuangping Huang. One-DM: One-shot diffusion mimicker for handwritten text generation. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 410–427, Cham, 2025a. Springer Nature Switzerland. ISBN 978-3-031-73636-0.

Gang Dai, Yifan Zhang, Yutao Qin, Qiangya Guo, Shuangping Huang, and Shuicheng Yan. Beyond isolated words: Diffusion brush for handwritten text-line generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19054–19064, October 2025b.

Ji Gan, Baijin Li, Yan-Ming Zhang, Jiaxu Leng, Weiqiang Wang, and Xinbo Gao. Stylized handwriting generation of arbitrary structures and OOV expressions: A decoupled approach via layout-offsets, 2025. URL https://openreview.net/forum?id=SuLp0J2uan.

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*, KDD '25, pp. 5459–5469, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714542. doi: 10.1145/3711896.3737436.

Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 369–376, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143891.

Tongkun Guan, Chengyu Lin, Wei Shen, and Xiaokang Yang. PosFormer: Recognizing complex handwritten mathematical expression with position forest transformer. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 130–147, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72670-5.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. https://bfl.ai/blog/flux-2, 2025.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space, 2025. URL https://arxiv.org/abs/2506.15742.

Bohan Li, Ye Yuan, Dingkang Liang, Xiao Liu, Zhilong Ji, Jinfeng Bai, Wenyu Liu, and Xiang Bai. When counting meets HMER: Counting-aware network for handwritten mathematical expression recognition. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 197–214, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19815-1.

Yu Li, Jin Jiang, Jianhua Zhu, Shuai Peng, Baole Wei, Yuxuan Zhou, and Liangcai Gao. UniMuMER: Unified multi-task fine-tuning of vision-language model for handwritten mathematical expression recognition. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=oHbVboLXz6.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

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, 2019. doi: 10.1109/ICDAR.2019.00247.

Bruce Miller. LaTeXML: a tex and latex to xml/html/epub/mathml translator. https://github.com/brucemiller/LaTeXML, 2026. Accessed: 2026-03-05.

H. Mouchère, C. Viard-Gaudin, R. Zanibbi, and U. 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, 2014. doi: 10.1109/ICFHR.2014.138.

Harold Mouchère, Christian Viard-Gaudin, Richard Zanibbi, and U. 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, 2016. doi: 10.1109/ICFHR.2016.0116.

Konstantina Nikolaidou, George Retsinas, Giorgos Sfikas, and Marcus Liwicki. DiffusionPen: Towards controlling the style of handwritten text generation. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 417–434, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-73013-9.

Wei Pan, Huiguo He, Hiuyi Cheng, Yilin Shi, and Lianwen Jin. DiffInk: Glyph- and style-aware latent diffusion transformer for text to online handwriting generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Vittorio Pippi, Silvia Cascianelli, and Rita Cucchiara. Handwritten text generation from visual archetypes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22458–22467, June 2023.

Vittorio Pippi, Fabio Quattrini, Silvia Cascianelli, Alessio Tonioni, and Rita Cucchiara. Zero-shot styled text image generation, but make it autoregressive. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7910–7919, June 2025.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=di52zR8xgf.

Minsi Ren, Yan-Ming Zhang, and yi chen. Decoupling layout from glyph in online chinese handwriting generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=DhHIw9Nbl1.

Douglas Reynolds. Gaussian mixture models. In *Encyclopedia of biometrics*, pp. 827–832. Springer, 2015.

Antonio Sclocchi, Alessandro Favero, Noam Itzhak Levi, and Matthieu Wyart. Probing the latent hierarchical structure of data via diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=0GzqVqCKns.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michael Ramamonjisoa, Francisco Massa, Daniel HAZIZA, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Herve Jegou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=2NlGyqNjns. Featured Certification.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Matthias Springstein, Eric Müller-Budack, and Ralph Ewerth. Unsupervised training data generation of handwritten formulas using generative adversarial networks with self-attention. In *Proceedings of the 2021 Workshop on Multi-Modal Pre-Training for Multimedia Understanding*, MMPT '21, pp. 46–54, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450385305. doi: 10.1145/3463945.3469059.

Qwen Team. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Z-Image Team. Z-Image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025b.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19681–19690, June 2025.

Yiming Wang, Hongxi Wei, Heng Wang, and Shiwen Sun. SFRD: Handwritten mathematical expressions generation by spatial-aware feature refinement diffusion. In Xu-Cheng Yin, Dimosthenis Karatzas, and Daniel Lopresti (eds.), *Document Analysis and Recognition – ICDAR 2025*, pp. 414–428, Cham, 2026. Springer Nature Switzerland.

Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16133–16142, June 2023.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15703–15712, June 2025.

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 (CVPR)*, pp. 4553–4562, June 2022.

Jianshu Zhang, Jun Du, Yongxin Yang, Yi-Zhe Song, Si Wei, and Lirong Dai. A tree-structured decoder for image-to-markup generation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 11076–11085, 2020.

Wenqi Zhao and Liangcai Gao. CoMER: Modeling coverage for transformer-based handwritten mathematical expression recognition. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 392–408, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19815-1.

Wenqi Zhao, Liangcai Gao, Zuoyu Yan, Shuai Peng, Lin Du, and Ziyin Zhang. Handwritten mathematical expression recognition with bidirectionally trained transformer. In Josep Lladós, Daniel Lopresti, and Seiichi Uchida (eds.), *Document Analysis and Recognition – ICDAR 2021*, pp. 570–584, Cham, 2021. Springer International Publishing. ISBN 978-3-030-86331-9.

Jianhua Zhu, Wenqi Zhao, Yu Li, Xingjian Hu, and Liangcai Gao. TAMER: Tree-aware transformer for handwritten mathematical expression recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(10):10950–10958, Apr. 2025. doi: 10.1609/aaai.v39i10.33190.

Algorithm 1 LaTeX to RelAST Conversion

Require: LaTeX expression L
Ensure: RelAST node T
$M \leftarrow \text{LaTeXML}(L)$ ▷ convert LaTeX to Presentation MathML
function BUILDRELAST(x)
  $tag \leftarrow \text{TAG}(x)$
  if tag is a symbol node (mi, mn, mo, $mtext$) **then**
    return CREATESYMBOLNODE(x)
  end if
  if tag denotes a linear container ($mrow$) **then**
    $C \leftarrow [\text{BUILDRELAST}(c) \mid c \in Children(x)]$
    return CONNECTSEQUENTIALLY($C, right$)
  end if
  if tag denotes a script structure ($msub$, $msup$, $msubsup$) **then**
    $base \leftarrow \text{BUILDRELAST}(Base(x))$
    for each script s in x **do**
      ATTACH($base$, BUILDRELAST(s), $ScriptRelation(s)$)
    end for
    return $base$
  end if
  if tag denotes a fraction or root **then**
    $T \leftarrow \text{CREATENODE}(tag)$
    for each child c in x **do**
      ATTACH(T, BUILDRELAST(c), $StructuralRelation(c)$)
    end for
    return T
  end if
  if $tag = mtable$ **then**
    return BUILDTABLETREE(x)
  end if
  $C \leftarrow [\text{BUILDRELAST}(c) \mid c \in Children(x)]$
  return CONNECTSEQUENTIALLY($C, right$)
end function
return BUILDRELAST($Root(M)$)

A THE PROPOSED DIFFMATH FRAMEWORK

A.1 RELAST CONSTRUCTION.

Given a LaTeX expression, we first convert it into Presentation MathML using the LaTeXML Miller (2026) toolkit, which exposes the structural hierarchy of mathematical expressions in a tree representation. Based on the resulting MathML structure, we recursively construct a relation-aware abstract syntax tree (RelAST), as described in Algorithm 1. Atomic symbols are mapped to leaf nodes, while sequential layouts are connected using the *right* relation to preserve reading order. Two-dimensional mathematical structures are explicitly represented by spatial relations, including *sup*, *sub*, *above*, *below*, and *inside*, corresponding to superscripts, subscripts, fractions, and roots. For matrix-like expressions, hierarchical row–cell structures are introduced to capture the two-dimensional layout.

To obtain a sequence representation suitable for generative modeling, the parsed structure is linearized via a depth-first traversal. Each node is represented as a tuple $[S, R, D]$, where S denotes the symbol, R the incoming relation, and D the tree depth. The resulting symbol–relation–depth sequence is defined as the RelAST representation, which preserves both symbol identities and structural dependencies of the original formula while providing a compact sequential format.

$(it)^r\psi(t)$


```
"latex": "(it)^{r}\\psi(t)",
"tree": ["<ROOT>", "(", "<RIGHT>", "i", "<RIGHT>", "t", "<RIGHT>", ")", "<SUP>",
"r", "<RIGHT>", "ψ", "<RIGHT>", "(", "<RIGHT>", "t", "<RIGHT>", ")"],
"depth": ["(: 0", "i: 0", "t: 0", "): 0", "r: 1", "ψ: 0", "(: 0", "t: 0", "): 0"]
```


$t_{ik} = \sum_j c_{ij} c_{jk}$


```
"latex": "t_{ik}=\\sum_{j}c_{ij}c_{jk}",
"tree": ["<ROOT>", "t", "<SUB>", "i", "<RIGHT>", "k", "<RIGHT>", "=", "<RIGHT>",
"Σ", "<SUB>", "j", "<RIGHT>", "c", "<SUB>", "i", "<RIGHT>", "j", "<RIGHT>", "c",
"<SUB>", "j", "<RIGHT>", "k"],
"depth": ["t: 0", "i: 1", "k: 1", "=: 0", "Σ: 0", "j: 1", "c: 0", "i: 1", "j: 1",
"c: 0", "j: 1", "k: 1"]
```


$(|a|^2, |b|^2, \ldots, |h|^2)$


```
"latex": "(|a|^{2},|b|^{2},...,|h|^{2})",
"tree": ["<ROOT>", "(", "<RIGHT>", "|", "<RIGHT>", "a", "<RIGHT>", "|", "<SUP>",
"2", "<RIGHT>", ",", "<RIGHT>", "|", "<RIGHT>", "b", "<RIGHT>", "|", "<SUP>", "2",
"<RIGHT>", ",", "<RIGHT>", "…", "<RIGHT>", ",", "<RIGHT>", "|", "<RIGHT>", "h",
"<RIGHT>", "|", "<SUP>", "2", "<RIGHT>", ")"],
"depth": ["(: 0", "|: 0", "a: 0", "|: 0", "2: 1", ",: 0", "|: 0", "b: 0", "|: 0",
"2: 1", ",: 0", "…: 0", ",: 0", "|: 0", "h: 0", "|: 0", "2: 1", "): 0"]
```


$(3)Z(p^\infty)$


```
"latex": "(3)Z(p^{\\infty})",
"tree": ["<ROOT>", "(", "<RIGHT>", "3", "<RIGHT>", ")", "<RIGHT>", "Z", "<RIGHT>",
"(", "<RIGHT>", "p", "<SUP>", "∞", "<RIGHT>", ")"],
"depth": ["(: 0", "3: 0", "): 0", "Z: 0", "(: 0", "p: 0", "∞: 1", "): 0"]
```


Figure 7: Examples of RelAST representations, illustrating how mathematical expressions are organized into compact structural descriptions.

A.2 MathVAE Architecture

MathVAE serves as the latent representation module for the latent diffusion Transformer, learning compact latent features while preserving reconstruction fidelity. It adopts a hierarchical 1D convolutional encoder–decoder with residual connections. The encoder consists of two strided convolutional layers, each followed by a four-layer residual stack, progressively compressing the input trajectory from $\mathbb{R}^{N\times 4}$ to a latent representation of $\mathbb{R}^{N/4\times 384}$. The decoder mirrors this architecture using residual stacks and transposed convolutions to progressively recover the temporal resolution. The reconstructed features are further processed by a Transformer-based trajectory decoder comprising a linear projection, a 4-layer Transformer encoder (hidden size 256, 4 attention heads), and an output projection, which predicts the parameters of a $p = 20$ component Gaussian Mixture Model (GMM Reynolds (2015)) for (x, y) trajectory generation, while the remaining dimensions are used for pen-state classification. To preserve semantic information in the latent space, we additionally employ an auxiliary OCR branch on the encoder features. It consists of a linear projection, sinusoidal positional encoding, and a 3-layer Transformer encoder (hidden size 256, 4 attention heads), followed by a linear classifier optimized with the CTC loss, encouraging the latent representations to retain character-discriminative information without explicit point-level alignment.

A.3 MathDiT Architecture

The overall training and inference procedures are summarized in Algorithm 2. MathDiT is trained to predict the clean latent representation x_0 from its noisy counterpart x_t, while inference progressively transforms Gaussian noise into the final latent sample. In implementation, MathDiT takes the noisy latent sequence, the RelAST-derived symbol and depth sequences, and the global symbol count N_c as conditions. Symbol and depth embeddings are first summed to form 512-dimensional token representations and contextualized by three stacked ConvNeXt-V2Woo et al. (2023) blocks, each consisting of a depthwise 1D convolution, Layer Normalization, a two-layer feedforward network with GELU activation and Global Response Normalization (GRN), and residual connections. The resulting structure-aware content features are concatenated with the latent input and projected to a 768-dimensional representation, followed by a convolutional positional embedding. The fused features are then processed by a 16-layer DiT backbone with multi-head self-attention, feedforward networks, rotary positional embeddings, and AdaLN-Zero modulation. To incorporate global expression complexity, the symbol count N_c is logarithmically normalized and used to modulate the

Algorithm 2 MathDiT Training and Inference

Training Phase:

Input: x_0, Z, N_c
Sample $t \sim \mathcal{U}(1, T)$
Sample noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$
$x_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$
$\hat{x}_0 = \text{MathDiT}_\theta([x_t, Z], t, N_c)$
Update MathDiT_θ: $\|\hat{x}_0 - x_0\|^2$.

Inference Phase:

Input: $x_T \sim \mathcal{N}(0, \mathbf{I}), Z, N_c$
for $t = T, \ldots, 1$ **do**
  $\hat{x}_0 = \text{MathDiT}([x_t, Z], t, N_c)$
  $\epsilon = (x_t - \sqrt{\bar{\alpha}_t} \cdot \hat{x}_0)/\sqrt{1 - \bar{\alpha}_t}$
  $x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \cdot \hat{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1}} \cdot \epsilon$
end for ▷ Final output: $\hat{x}_0$

magnitude of the timestep embedding through a learnable time-dependent scaling factor, providing stronger guidance during later denoising stages. Finally, the modulated features are projected back to the latent space to predict the clean latent representation.

B Implementation Details

B.1 Proposed DiffMath

DiffMath is implemented using the PyTorch framework. Specifically, **MathVAE** is trained on a single NVIDIA H800 GPU with a batch size of 128 for 100 epochs. The training uses FP32 precision and the AdamW optimizer. The initial learning rate is set to 1.5×10^{-4} with a minimum learning rate of 1×10^{-5}, and a warmup schedule of 5,000 steps is adopted. In the loss formulation, the weights of L_{gmm}, L_{pen}, and L_{per} are set to 0.1, 1.0, and 1.0, respectively.

MathDiT is trained on 8 NVIDIA H800 GPUs with a per-GPU batch size of 128 for approximately 220k training steps. The model also uses FP32 precision and the AdamW optimizer. The initial learning rate is set to 3×10^{-4} with a minimum learning rate of 3×10^{-6}, together with a warmup of 20k steps. The diffusion process uses 1000 timesteps with a cosine noise schedule. During training, the conditioning input is randomly dropped with a probability of 10% to enable classifier-free guidance (CFG Ho & Salimans (2021)). During inference, the guidance scale is set to 3.0. DDIM Song et al. (2021) sampling is adopted for acceleration, with the number of sampling steps set to 20.

B.2 Baseline Models

For comparison, we include several representative generation models, including Formulagan Springstein et al. (2021), One-DM Dai et al. (2025a), SD-XL Podell et al. (2024), FLUX.1 Labs (2025), Z-Image Team (2025b), Qwen-Image Wu et al. (2025), and DiffInk Pan et al. (2026). Among them, DiffInk, similar to our method, performs generation in the online sequence modality. Formulagan and One-DM are specialized models designed for formula generation, while SD-XL, FLUX.1, Z-Image, and Qwen-Image are general text-to-image generation models. Below we describe the training and inference details.

For the early GAN-based generation method FormulaGAN, we directly use the pretrained weights released by the authors for inference. For the handwritten text-line generation method DiffInk, we replace the original text input with LaTeX expressions as the conditioning input and remove the style branch in the VAE module, retaining only the content-related regularization. The remaining training and testing settings follow the default configuration. During inference, the CFG scale is set to 3.0 with 20 sampling steps.

For the dedicated handwriting generation model One-DM, we remove the style branch from the original method while keeping the remaining settings the same as the default configuration during training. During inference, we set the CFG scale to 4 to achieve better generation quality.

For general-purpose text-to-image models, including Qwen-Image, Z-Image, SD-XL, and FLUX.1, we fine-tune the models using LoRA with rank = 64 for 5 epochs. During inference, the CFG scale is set to 5 for Qwen-Image and Z-Image, 5.0 for FLUX.1, and 7.5 for SD-XL to obtain better generation results.

B.3 OCR MODELS

For the OCR models, we adopt Qwen-3VL-8B Instruct Team (2025a) and Uni-Mumer Li et al. (2025) (built upon Qwen-2.5VL-3B) as the backbone models. When training with the original data as well as the dataset augmented with DiffMath-generated samples, we perform full-parameter fine-tuning. For both the MathWriting Gervais et al. (2025) and CROHME Mouchère et al. (2014; 2016); Mahdavi et al. (2019) datasets, the models are fine-tuned for one epoch.

B.4 DETAILS OF METRICS

For evaluation, we adopt four metrics: Expression Recognition Rate (ExpRate), Character Detection Matching (CDM), ExpRate@CDM (@CDM Wang et al. (2025)), and a User Study. ExpRate measures the string-level accuracy between the predicted and ground-truth LaTeX expressions. Since string matching alone cannot fully reflect visual structural correctness, following Uni-Mumer Li et al. (2025), we further adopt a more robust visual metric, CDM, which renders LaTeX expressions into images and evaluates consistency by detecting symbols and comparing their spatial layouts. ExpRate@CDM is a stricter criterion that requires the rendered results to be visually identical, reducing errors caused by different writing styles or equivalent layouts. In addition, we conduct a user study for subjective evaluation. We randomly sample 200 LaTeX expressions from the test set and generate the corresponding handwritten formulas using each competing method. Fifteen volunteers are invited to participate in the evaluation. For each expression, participants are presented with the generated results from all methods and asked to select the one they consider the best, taking into account both the correctness of symbols and structural layout as well as the naturalness of the handwriting style. The user study score is then computed as the proportion of votes received by each method across all evaluations.

C MORE EXPERIMENTAL RESULTS

C.1 COMPARISON WITH BASELINE METHODS.

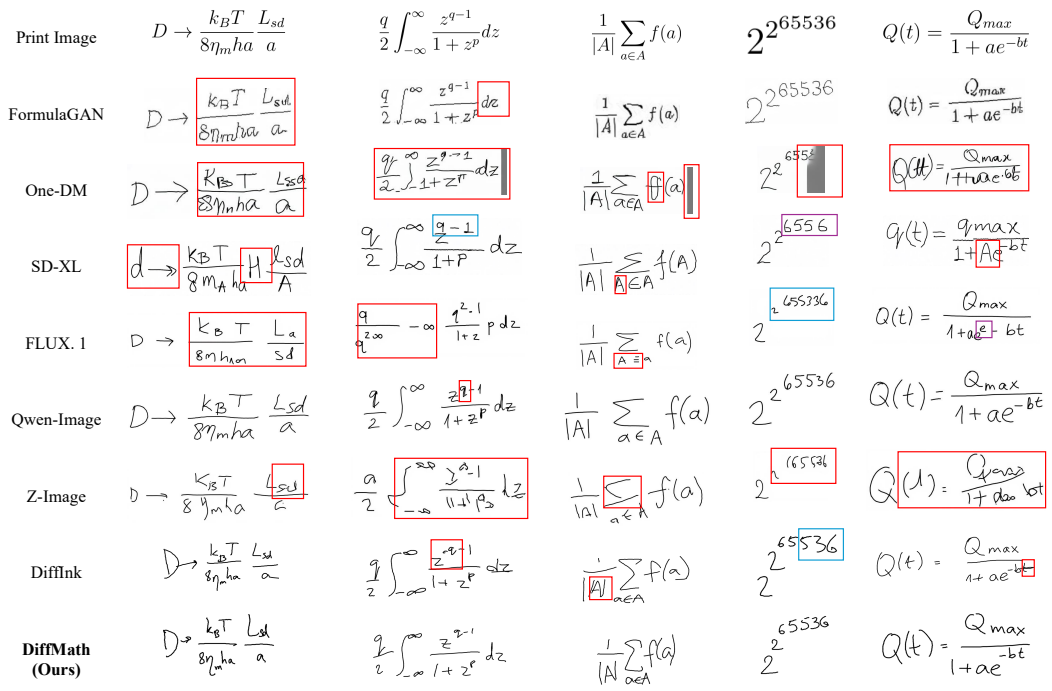
As shown in Fig.8(a), DiffMath consistently generates handwritten mathematical expressions with more accurate symbol placement and stronger structural coherence than baseline generation methods. Even for formulas containing nested superscripts, fractions, roots, and matrix-like layouts, our method better preserves the intended mathematical structure while producing visually natural handwriting. Figure8(b) further showcases more challenging and diverse mathematical expressions generated by DiffMath, highlighting its ability to synthesize complex two-dimensional layouts with high structural fidelity and robustness.

C.1.1 MORE RESULTS GENERATED BY DIFFMATH.

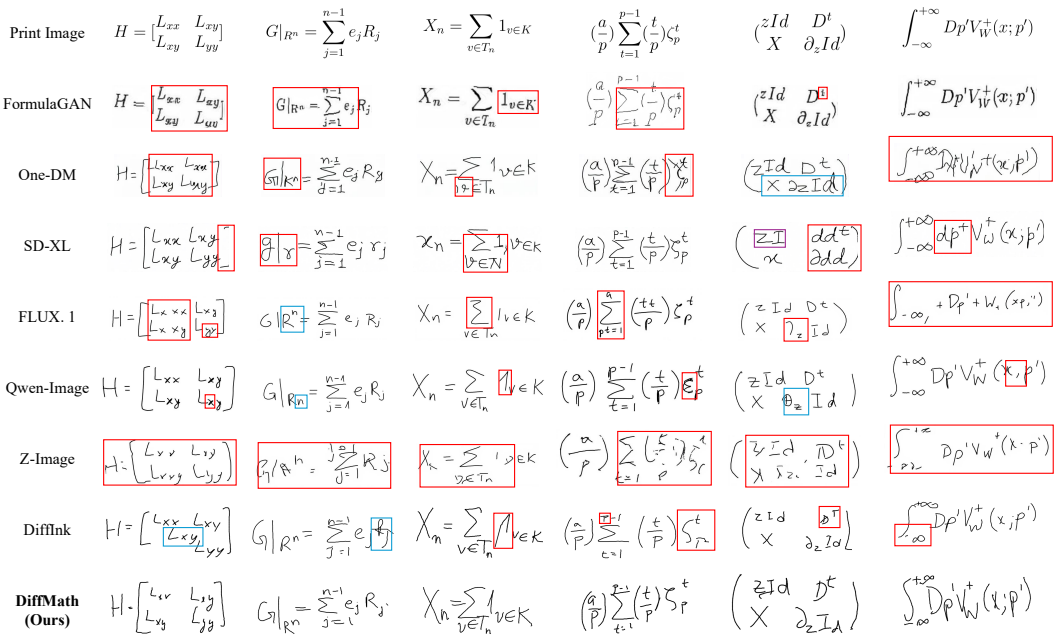
Furthermore, we present additional examples generated by DiffMath, as shown in Fig. 9. The upper part contains matrix expressions in various formats, while the lower part includes more complex expressions with numerous symbols and fine-grained details that are prone to errors. Even in these challenging cases, DiffMath maintains high accuracy in both symbol content and structural consistency.

C.2 FAILURE CASE ANALYSIS

Fig. 10 presents representative failure cases. Most errors occur in LaTeX expressions with rare symbols, dense layouts, or deep nesting, where the model may omit small components, confuse similar symbols, or misplace superscripts, subscripts, and fraction elements. These cases indicate that compact and complex mathematical structures remain challenging for generation.



(a) Comparison on **simple and common** expressions.



(b) Comparison on **more complex** expressions.

Figure 8: **More qualitative comparisons with SOTA methods.** Red boxes highlight atomic content errors (incorrect symbols or artifacts), blue boxes indicate structural misalignments, and purple boxes denote missing symbols. **DiffMath (Ours)** demonstrates strong performance in **both simple and complex LaTeX scenarios**, maintaining high fidelity in symbol content and 2D topology.

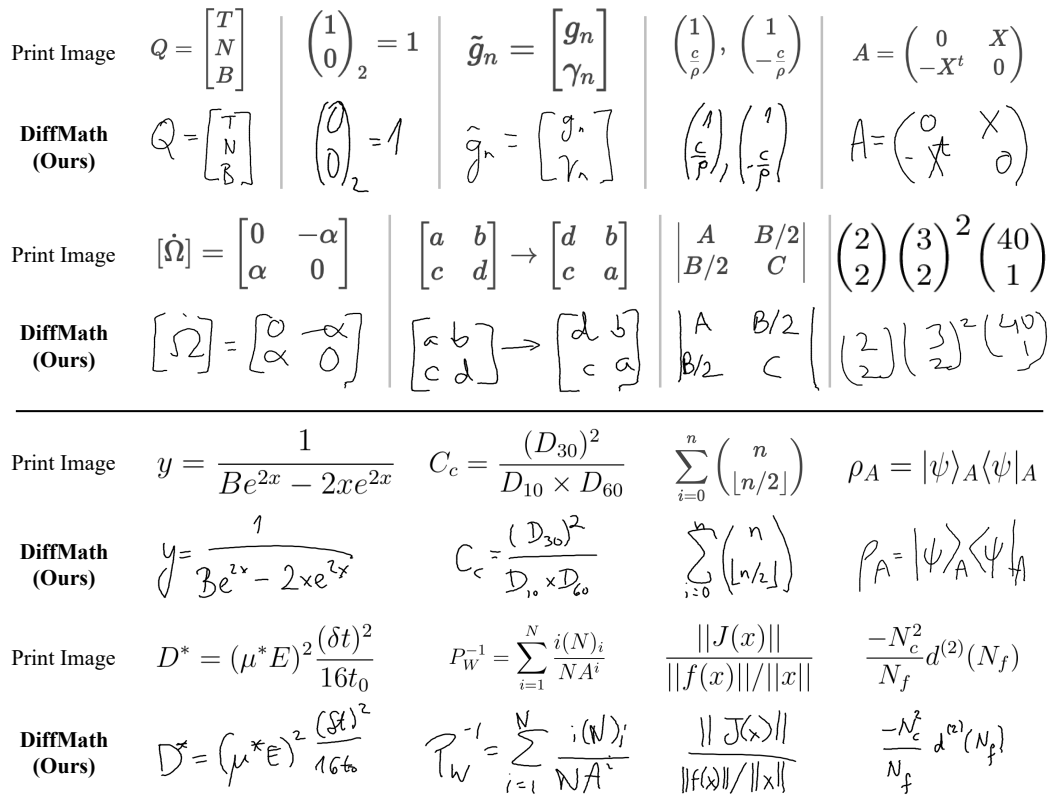


Figure 9: **Additional samples generated by DiffMath (Ours).** The upper part shows matrix expressions, while the lower part presents long formulas with dense symbols and complex local layouts. DiffMath consistently generates complex mathematical expressions with accurate symbols and correct structural relationships.

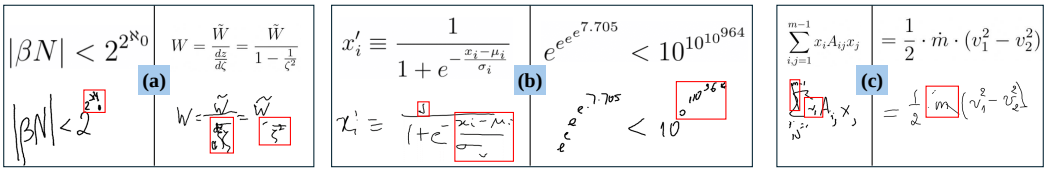


Figure 10: Failure cases of generated samples: (a) rare symbols, (b) complex nested structures, and (c) locally dense text regions.