Generating Handwritten Mathematical Expressions From Symbol Graphs: An End-to-End Pipeline

Yu Chen[1,*], Fei Gao[2,*], Yanguang Zhang[3], Maoying Qiao[4], Nannan Wang[5,†]
[1] Beijing Waiyan Online Digital Technology; [2] Hangzhou Institute of Technology, Xidian University
[3] Hangzhou Dianzi University; [4] University of Technology, Sydney (UTS); [5] Xidian University
{fgao, nnwang}@xidian.edu.cn, ppak1991@outlook.com, maoying.qiao@uts.edu.au

Abstract

In this paper, we explore a novel challenging generation task, i.e. Handwritten Mathematical Expression Generation (HMEG) from symbolic sequences. Since symbolic sequences are naturally graph-structured data, we formulate HMEG as a graph-to-image (G2I) generation problem. Unlike the generation of natural images, HMEG requires critic layout clarity for synthesizing correct and recognizable formulas, but has no real masks available to supervise the learning process. To alleviate this challenge, we propose a novel end-to-end G2I generation pipeline (i.e. $graph \rightarrow layout \rightarrow mask \rightarrow image$), which requires no real masks or nondifferentiable alignment between layouts and masks. Technically, to boost the capacity of predicting detailed relations among adjacent symbols, we propose a Less-is-More (LiM) learning strategy. In addition, we design a differentiable layout refinement module, which maps bounding boxes to pixel-level soft masks, so as to further alleviate ambiguous layout areas. Our whole model, including layout prediction, mask refinement, and image generation, can be jointly optimized in an end-to-end manner. Experimental results show that, our model can generate high-quality HME images, and outperforms previous generative methods. Besides, a series of ablations study demonstrate effectiveness of the proposed techniques. Finally, we validate that our generated images promisingly boosts the performance of HME recognition models, through data augmentation. Our code and results are available at: https://github.com/AiArt-HDU/HMEG.

1. Introduction

Handwritten Mathematical Expressions (HMEs) are common and play significant roles in our daily life, especially in the research and education areas. HMEs generally present complex structures, serious deformations, and diverse writing styles. Such characteristics, along with data scarcity, make HME Recognition (HMER) a grand challenge in the OCR community. Despite the tremendous efforts that have been made to this task, the HMER performance is still unsatisfactory [14, 27, 62].

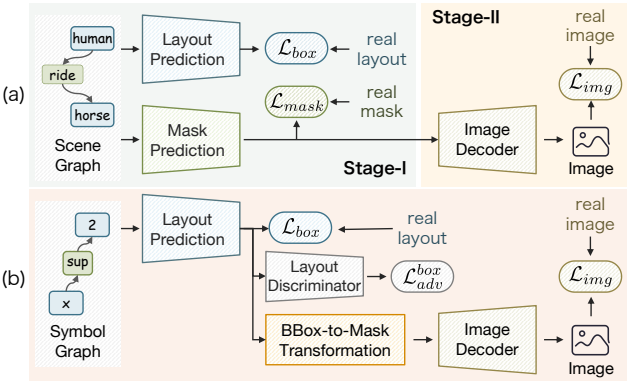

Figure 1. Differences between (a) typical two-stage graph-to-image generation pipeline and (b) our end-to-end pipeline. In previous methods, the real masks are available as the input. In contrast, we propose a novel end-to-end pipeline of $graph \rightarrow layout \rightarrow mask \rightarrow image$, and requires no real masks or *non-differentiable* alignment between layouts and masks.

Recently, synthetic data augmentation has shown inspiring performance in various recognition tasks [23, 51]. This is mainly achieved by the remarkable progress of conditional image generation [21] in the past few years [58]. The advances allow for creating realistic images through texts [39, 57, 66], scene graphs [24, 60], layouts [28, 69], semantic masks [37, 72], sketches [61, 67], and more [12, 65]. However, there has been no effective generative model for generating high-quality HME images.

Some previous works synthesize HMEs by recomposing real online HMEs [46, 59]. Besides, FormulaGAN [41] generates HMEs from rendered images, via an *Image-to-Image* (I2I) Translation model. However, their generated images are limited in either diversity or realism. Be-

[*] Equal Contributions. [†] Corresponding Author.

sides, *Mathematical Expressions* (MEs) are highly structured data. Given an HME, its meaning and handwritten style, depend not only on the category and shape of each symbol, but also on the positional relations between symbols. It's unlikely to design an elaborate I2I translation model that can handle all these issues.

To combat this challenge, in this paper, we explore a new generation task, i.e. *Handwritten Mathematical Expression Generation* (HMEG) *from symbol graphs of symbolilc sequences*. It's natural and easy to use a graph, specifying symbols and positional relations between symbols, to represent an ME [63]. Besides, previous works have demonstrated the significance of fully mining the structural information in boosting generalization [52]. By converting symbolic sequences to symbol graphs, we propose a novel HMEG method, inspired by the classic graph-to-image (G2I) generation work, Sg2im [24]. Similar to existing G2I generation works, our method consists of three stages: graph-to-layout prediction, layout-to-mask transformation, and mask-to-image generation.

Compared to the generation of natural images, the key challenge of HMEG is to generate *unambiguous* structural layouts, for accurately presenting meanings of formulas. From one hand, the requirement of layout clarity is much critical for generating formulas. Not only the size of each symbol, but also the positional relations between symbols, should be precisely predicted. Otherwise, mathematical meanings might change or become unrecognisable. For example, an input formula x^2 might become x_2 or $x2$ in a generated image. From the other hand, MEs have infinite possible layouts and structures. Namely, a formula could comprise of arbitrary number of symbols and arbitrary relations between symbols. It's impossible to collect completed training data for learning the layout predictor.

To alleviate these difficulties, we first propose a Less-is-More (LiM) training strategy for learning an effective layout predictor. Specially, our layout predictor follows the structure of Graph Convolutional Networks (GCNs) [30]. During training, we only use minimal graphs with 1-degree connections, in stead of whole graphs with complex structures, to optimize the layout predictor. Note that the learned predictor can be applied to arbitrary symbol graphs, in the testing stage. By using LiM, the predictor emphasizes on local structures between adjacent symbols. Besides, the learning process is eased. Namely, we could learn an effective layout predictor by using moderate amount of training samples. Such benefits will be verified in the ablation study part. Besides, we add a layout discriminator to further boosts the realism of predicted layouts.

In addition, we propose a *Sequential BBox-to-Mask Transformation* (B2M) module, for refining layouts under weak supervision. In the original layout, bounding boxes of symbols tend to overlap with each other. Such overlapped areas may lead to overlapping symbols or chaotic strokes, in generated images. It's critical to refine coarse layouts to pixel-level soft masks, to alleviate the overlapping problems [22]. Most previous G2I generation works require real segmentation masks for learning refined layout masks [13, 22]. However, it's difficult to define the mask of a symbol, because white pixels around strokes are essential for representing a symbol. We have to refine the layout under weak supervision. To this end, we propose to learn a sequential mapping of $layout \rightarrow grid \rightarrow mask$, and use an subsequent image decoder to generate an HME image. During training, we use the image-related losses to train both B2M and the image decoder jointly.

In summery, we propose a novel end-to-end pipeline of $graph \rightarrow layout \rightarrow mask \rightarrow image$ for G2I generation tasks. Our model requires no supervision of real masks, or interactive alignment operations between layouts and masks. The major differences between our method and previous typical G2I generation works, are illustrated in Fig. 1. To verify the effectiveness of the proposed techniques, we conduct experiments on CROHME2014/2016/2019 [34]. Results show that, our model can generate high-quality HME images with clear layouts and recognisable symbols. Besides, our model significantly outperforms a number of existing generative methods, both qualitatively and quantitatively. A series of ablation study demonstrate the effectiveness of the proposed techniques, including the LiM, layout discriminator, and B2M. Finally, our generated HME images prove to boost the HMER performance by 5-11% absolutely, through synthetic data augmentation.

2. Related Works

HME Recognition (HMER). Most advanced methods treat HMER as an image-to-markup problem [53]. They typically use *Convolutional Neural Networks* (CNNs) or *Recurrent Neural Networks* (RNNs) [18] to encode input images; and use attention-based models, such as *Gated Recurrent Units* (GRU) [5, 14, 68], or Transformers [2, 47, 70], to generate LaTeX sequences. Besides, some methods have been proposed [41, 46, 59]. Some recent works propose to use additional information, e.g. symbol-level counting [27], or relation-level counting [14], or emphasis on visually similar symbols [33], or synthetic data augmentation, to boost the HMER performance. Another branch of HMER are graph-based methods. These methods typically transform LaTeX sequences to *Symbol Label Trees* (SLTs) [63], and use tree-structured decoders to obtain the outputs [14, 52, 54]. These works validate the capacity of graphs in representing the structural information of formulas. In this paper, we therefore convert symbolic sequences to graphs, and use a GCN to encode them for generating HME images.

Graph-to-Image Generation. Image generation from scene graphs is first proposed in Sg2im [24]. The pipeline

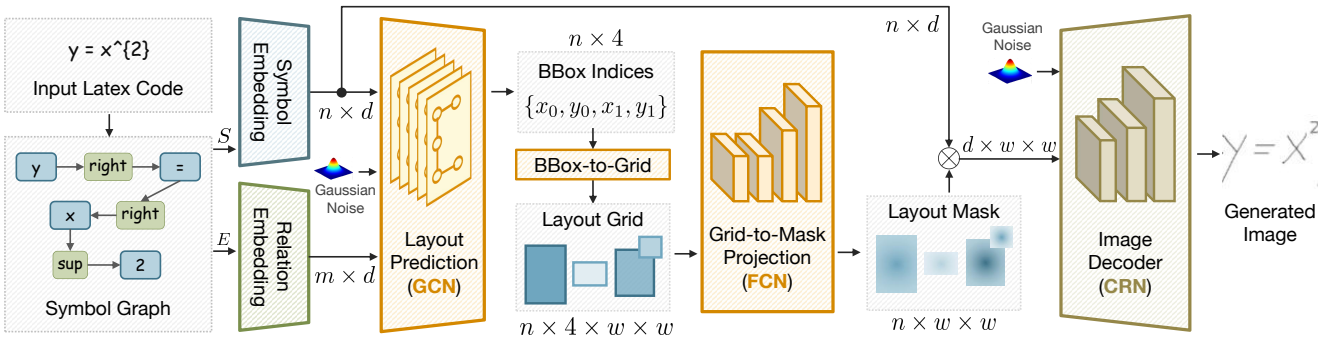

Figure 2. Overview of our handwritten mathematical expression generator (HMEG).

of Sg2im includes two stages: *graph-to-layout* mapping and *layout-to-image* generation. Various subsequent methods have been proposed to boost the quality of generated images, following the same pipeline [9, 13, 16, 40]. Besides, some semi-parametric methods use crops from real images to boost the quality of generated images [31]. There are also some extensions to image manipulation [9, 42].

Most of these methods use a image discriminator and a object discriminator during training. Besides, they use real layouts, segmentation masks, and images, to supervise the learning process [13, 22]. COLoR [22] additionally uses a mask discriminator to boost the precision of generated masks. Recently, KCGM [55] uses knowledge consensus to alleviate the dependency on real layouts or masks. Besides, several diffusion model based methods [12, 60] can generate high-quality images directly from input scene graphs. However, diffusion based methods still suffer the high cost during both training and inference. Besides, generating recognisable texts is still challenging for most diffusion models.

Layout-to-Image Generation. Most layout-to-image generation methods separately predict the mask of each object based on object embeddings, and require frequent resizing interaction between bounding boxes and masks [43, 44, 69]. To improve the quality of generated masks, several methods use masks discriminator [1, 28], and requires real masks during training. Besides, OC-GAN [45] uses both global and local similarities between scene graphs and images are used to improve the realism of generated layouts. Similar to our method, LAMA [32] tries to alleviate the overlapping between objects. However, in all these methods, the real bounding boxes are available as the input. They at most implement the end-to-end computing of $layout \rightarrow mask \rightarrow image$. In contrast, we propose a novel end-to-end computing of $graph \rightarrow layout \rightarrow mask \rightarrow image$, and requires no interactive alignment operations between layouts and masks.

Handwritten Font Generation. Another related task is *Handwritten Font Generation* (HFG), which aims at transferring images of fonts to the target handwritten style [48, 50]. Most methods follow the I2I translation pipeline, and adopt encoder-decoder architectures [6]. Besides, great efforts have been made to solve the few-shot learning challenge [48, 64]. These methods focus on the style of strokes. In contrast, our work aims to generate rational layouts of multiple symbols, with complex relations, at the same time. It's interesting to explore generating HME images in personal styles in the future.

3. Method

3.1. Overview

Our goal is to generate HME images conditioned on LaTeX sequences. To this end, we represent LaTeX sequences by symbol graphs [63] and design a novel G2I generation method to synthesize HME images. Our full model follows the idea of Generative Adversarial Networks (GANs) [21, 24]. It includes a generator, and several discriminators in the training stage. Fig. 2 shows the pipeline of our generator. It mainly include four parts: (i) First, the input LaTeX sequence is converted to a symbol graph, and embedded into high-dimensional feature vectors. (ii) A GCN based layout predictor estimates the coarse layout, i.e. bounding box (BBox) of each symbol. (iii) A sequential layout refinement module, including *BBox-to-Grid* mapping and *Grid-to-Mask* projection, generate a pixel-level soft mask for each symbol. (iv) The image decoder, a Cascaded Refinement Network (CRN) [4], generates an image conditioned on the predicted mask.

3.2. Symbol Graph Construction

Given a set of mathematical symbol categories $\mathcal{C}$ and a set of relation categories $\mathcal{R}$, a symbol graph (S, E) can be constructed based on $\mathcal{C}$ and $\mathcal{R}$: each node i in $S = \{s_1, \cdots, s_n\}$ is a mathematical symbol, with $s_i \in \mathcal{C}$; and each directed edge (i, j) in $E = \{e_{ij}\}$ represents the positional relation of node j w.r.t. node i, with $e_{ij} = (s_i, r, s_j)$ and $r \in \mathcal{R}$. n is the number of symbols in a mathematical

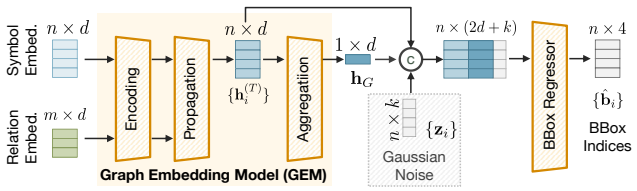

Figure 3. Architecture of our layout predictor and the Graph Embedding Model (GEM) [30].

expression. Let m denote the number of edges in E. For example, given an expression x^2, it can be represented by a tuple $(x, sup, 2)$, where sup is short for $superscript$. Besides, we add an additional edge type `None` to indicate that the corresponding pair of symbols are unrelated.

Afterwards, we map the nodes and edges (S, E) to high-dimensional feature vectors $\{\mathbf{V}, \mathbf{U}\}$ by using a learnable embedding layer, respectively [30]. The learned embedding feature vectors are denoted by:

$$\mathbf{V} = \{\mathbf{v}_i, \forall i \in S\}, \mathbf{U} = \{\mathbf{u}_{ij}, \forall (i,j) \in E\}, \quad (1)$$

where $\mathbf{v}_i, \mathbf{u}_{ij} \in \mathbb{R}^d$ associate with s_i, e_{ij}, respectively; d is the dimension of embedding vectors.

3.3. Adversarial Layout Prediction

Inspired by GAN [21], we train our layout prediction module in an adversarial learning manner. Specially, we use an layout predictor G_{box} to generate a layout in the format of bounding boxes, conditioned on a given symbol graph; and use an additional layout discriminator D_{box} to judge whether a layout is real or fake. Using the adversarial learning strategy like GAN, the layout predictor would finally predict high-quality bounding boxes.

Layout Predictor. We first use the Graph Embedding Model (GEM) [30] to update features of nodes and edges, and then use a MLP to predict the BBox indices of each symbol. First, original embedding features are encoded by: $\mathbf{h}_i^{(0)} = f_{\text{node}}(\mathbf{v}_i), \forall i \in S$ and $\mathbf{e}_{i,j} = f_{\text{edge}}(\mathbf{u}_{ij}), \forall (i,j) \in E$. Afterward, node features are updated through a stack of propagation layers, by accumulating information in local neighborhood. Features of node i at the t-th layer are updated by:

$$\begin{aligned} \mathbf{m}_{j \to i} &= f_{\text{message}}(\mathbf{h}_i^{(t)}, \mathbf{h}_j^{(t)}, \mathbf{e}_{ij}), \forall (i,j) \in E, \\ \mathbf{h}_i^{(t+1)} &= f_{\text{node}}(\mathbf{h}_i^{(t)}, \sum_j \mathbf{m}_{j \to i}). \end{aligned} \quad (2)$$

After T rounds of propagations, the model produces an aggregated graph level representation by:

$$\mathbf{h}_G = f_G \left(\sum_i \sigma(f_{\text{gate}}(\mathbf{h}_i^{(T)})) \odot f_{\text{update}}(\mathbf{h}_i^{(T)}) \right), \quad (3)$$

where σ denotes the Sigmoid operation. Finally, we integrate the updated embedding features and the graph level features for predicting the BBox indices of each symbol. Besides, we concatenate a Gaussian noise vector $\mathbf{z}_i \in \mathbb{R}^k, \forall i \in S$ to these features, to improve the diversity in predicted layouts. The BBox prediction is formulated by:

$$\hat{\mathbf{b}}_i = f_{\text{bbox}}(\mathbf{h}_i^{(T)}, \mathbf{h}_G, \mathbf{z}_i), \forall i \in S. \quad (4)$$

Each of the mapping functions f_* above is an MLP on the inputs [29, 30]. For each symbol, we obtain a 4-dimensional rectangular box $[x_0, y_0, x_1, y_1]$ in the $[0, 1]$ coordinate space, denoting the coordinates of the upper left and lower right corners of the BBox border.

Layout Discriminator. The layout discriminator D_{box} is also composed of GEM [30], followed by an MLP to predict whether an input layout as real or fake. D_{box} takes concatenations of the symbol embedding matrix $\mathbf{V} \in \mathbf{R}^{n \times d}$ and the layout indices $B \in \mathbf{R}^{n \times 4}$ as input. In the implementation, we use the least square version of adversarial loss, i.e.

$$\mathcal{L}_{adv}^{box} = ||D_{box}(\mathbf{V}, B)||_2^2 + ||1 - D_{box}(\mathbf{V}, \hat{B})||_2^2. \quad (5)$$

Besides, we use the L2 distance between the predicted BBox indices $\hat{B}$ and the target (real) indices B as the layout reconstruction loss, i.e. $\mathcal{L}_{box} = ||B - \hat{B}||_1$. We use a weighted sum of $\mathcal{L}_{box}$ and $\mathcal{L}_{adv}^{box}$ as the objective of G_{box}. During training, G_{box} and D_{box} are alternatively optimized.

Less-is-More Learning (LiM) Strategy. A mathematical expression could includes arbitrary number of symbols or relations between symbols. In other words, there are infinite possible structures of symbol graphs. It's difficult to collect a complete training set. Besides, using complicated graphs during training may add the difficulty in learning an effective model. We therefore propose a Less-is-More (LiM) learning strategy [3]. Specially, we propose to use symbol graphs with merely 1-degree connections during training. In this way, the layout predictor is expected to emphasize on local structures, i.e. generating rational layout for locally connected symbols. While in the testing stage, the learned model can be applied to arbitrary symbol graphs, with arbitrary degrees of connections, and predict the corresponding layout.

3.4. Sequential BBox-to-Mask Transformation

We next predict the pixel-level soft mask of an expression based on the predicted bounding boxes. This procedure is illustrated in Fig. 4, and consists of the following two steps:

(1) *BBox-to-Grid Mapping.* We first transform the BBox indices to layout grids, as illustrated in Fig. 4. To this end, we subscribe each index value from the coordinate matrices $\mathbf{X}$ and $\mathbf{Y}$ in the $[0, 1]$ coordinate space. For each symbol, we obtain a $4 \times w \times w$ tensor of grid.

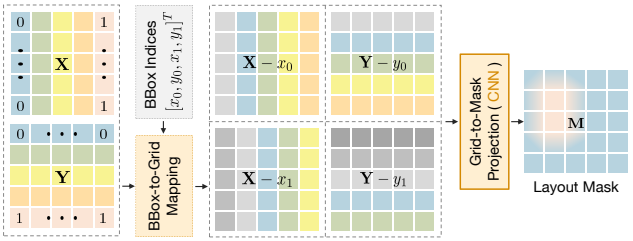

Figure 4. Illustration of the sequential BBox-to-Mask transformation (B2M) module.

(2) *Grid-to-Mask Projection.* Afterwards, we use a FCN with a Sigmoid output layer (denoted by ϕ_{mask}) to estimate a one-channel mask for each symbol, i.e.

$$\mathbf{M}_i = \phi_{\text{mask}}(\mathbf{X} - x_{i,0}, \mathbf{X} - x_{i,1}, \mathbf{Y} - y_{i,0}, \mathbf{Y} - y_{i,1}). \quad (6)$$

Each value in $\mathbf{M}_i$ indicates the probability of a pixel belonging to symbol s_i. Specially, we first use two Convolutional layers to predict a low-resolution mask of dimension $w \times w$, and then use two upsampling Convolutional layers to sequentially upsampling the mask by a factor of 2. These masks are used in the image decoder for generating an image of the same size, respectively.

Note: Previous G2I generation works [9] also consists of BBox-to-Grid mapping, but they merely use the grids to select patches of objects. Besides, they predict the mask from the BBox indices $\hat{B}$ or original embedding features $(\mathbf{V}, \mathbf{U})$. Differently, we use the transformed grids for layout mask prediction. Moreover, since the whole BBox-to-Mask prediction module is differentiable, our whole model enables training in an end-to-end manner.

3.5. Image Generation

Image Decoder. In the implementation, we adopt CRN [4] as the image decoder. The input is a multiplication of the whole layout mask $\mathbf{M} \in \mathbb{R}^{n \times w \times w}$ with the embedding vectors $\mathbf{V} \in \mathbb{R}^{n \times d}$. Besides, we add Gaussian noise to the feature maps to improve the diversity in image generation. CRN includes three upsampling Convolutional layers, and outputs a formula image with resolutions of $w \times w$. Recall that we actually predicts three masks at different scales. In the implementation, we have three generated images, at the resolution of 64×64, 128×128, and 256×256, respectively. All these images are used to calculate the losses for optimizing our model.

Image and Symbol Discriminators Finally, we use an image discriminator D_{img} and a symbol discriminator D_{sym} during training, to boost the realism of the generated whole image $\hat{I}$ and patches of symbols, respectively. We use the grids to select patches of symbols, and input them into D_{sym}. Both discriminators are fully CNNs with the cross-entropy (CE) version of adversarial loss. In addition, following AC-GAN [36], we have the symbol discriminator additionally predict category labels of symbols. In this way, the generated symbols would be more recognizable and real.

Loss Functions. Similar to previous G2I generation works [24], we use a series of loss functions to optimize the image generation. The losses are briefly introduced below.

- *Pixel loss* $\mathcal{L}_{pix}$*:* First, we use the L1 distance between a generated image and the target formula image and as the pixel loss: $\mathcal{L}_{pix} = ||I - \hat{I}||_1$.
- *Image adversarial loss* $\mathcal{L}_{adv}^{img}$*:* Second, we use the global image discriminant loss from D_{img}, to encourage the generated HME images in the real handwritten style.
- *Symbol adversarial loss* $\mathcal{L}_{adv}^{sym}$*:* Third, we use the symbol discriminative loss from D_{sym}, to improve the fidelity of generated symbols.
- *Auxiliary Symbol Recognition Loss* $\mathcal{L}_{aux}^{sym}$*:* Finally, we use the auxiliary symbol recognition loss from D_{sym}, to enforce the generator producing recognizable symbols.

We use a combination of these loss functions above, to train the B2M module and the image generation modules together. The total loss is formulated by:

$$\mathcal{L}_{img} = \lambda_1 \mathcal{L}_{pix} + \lambda_2 \mathcal{L}_{adv}^{img} + \lambda_3 \mathcal{L}_{adv}^{sym} + \lambda_4 \mathcal{L}_{aux}^{sym}, \quad (7)$$

where $\lambda_i, i = 1, ..., 4$ are weighting factors and are set as 1, 0.01, 0.01, 0.001, respectively.

4. Experiments

4.1. Experimental Settings

Data. We use CROHME2014/2016/2019 [34] to validate our HMEG method in terms of both generalization and recognition. The dataset collects pen trajectories, and labels them with symbols and symbol categories. The dataset contains 126 different symbol categories, and 8 positional relations, i.e. *start*, *left superscript*, *superscript*, *subscript*, *below*, *above*, *right* and *end*. By restoring the pen trajectory information, we can easily obtain the HME image and the BBox of each symbol. We use the standard partition of each dataset through all the experiments.

Implementation Details. In the implementation, due to the limitation of our computing resources, we train our model in two stages: we first train the layout prediction part in the first stage, and then train the rest parts by fixing it. In the first stage, we train the model with a batch size 64 and a learning rate 5e5, for 40,000 iterations. In each iteration, we random sample a batch of 1-degree sub-graphs from the training set for learning the layout predictor. In the second stage, we use a batch size 8, a learning rate 1e4, and train for 600,000 iterations. Our codes are implemented by using Pytorch and a NVIDIA TITAN XP GPU. We use Adam [26] as the optimizer for all networks.

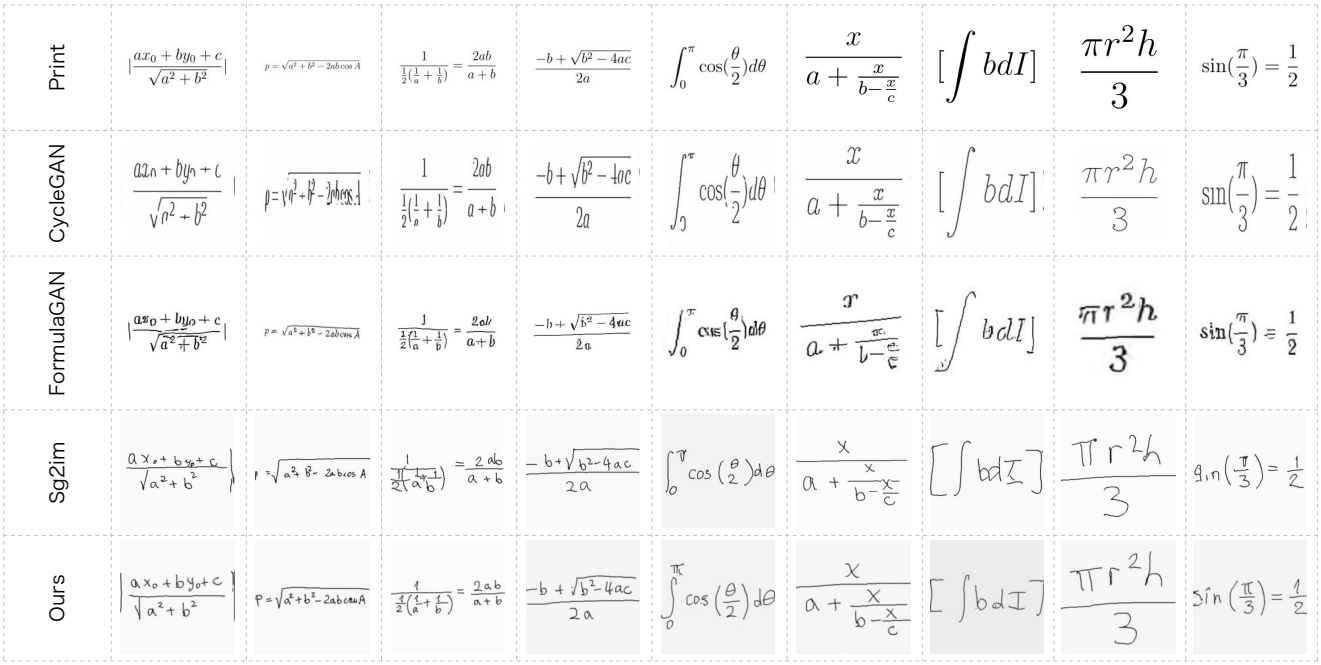

Figure 5. Handwritten mathematical expressions generated by CycleGAN [71], FormulaGAN [41], Sg2im [24], and our method.

Criteria. We choose two image quality assessment indices as the criteria. The former includes the *Fréchet Inception distance* (FID) [17] and *Structural Similarity Measure* (SSIM) [49]. Lower values of FID indicate better realism of synthesised images; while greater values of SSIM generally indicate higher similarity between a synthesized image and the corresponding real image. We here report the average SSIM value of all the test samples. In addition, we use two HMER indices, including the *Word Error Rate* (WER) and the *Expression Recognition Rate* (ExpRate). Here we use an open-source HMER method [19], to recognize HME images and compute the corresponding indices. Lower WER and higher ExpRate values indicate better performance.

4.2. Comparison with SOTAs

We first compare with the existing GAN based HMEG method, FormulaGAN [41], our baseline, Sg2im [24], and the widely used unsupervised I2I translation method, CycleGAN [71], on CHROME2019. We train and test these methods following the same experimental settings as ours.

Qualitative Comparison. Fig. 5 shows HME images generated by different models. CycleGAN fails to transform input formulas to the handwritten style. FormulaGAN produces handwritten-style images, but with blurring and unnatural strokes. In contrast, both Sg2im and our method generate high quality images, with realistic styles and clear strokes. In addition, the images generated by our method present better layouts than those by Sg2im. Sg2im sometimes produce very small symbols, e.g. y_0 in the first column and p in the second column. Besides, Sg2im may produce overlapped symbols, e.g. $\frac{1}{2}(\frac{1}{a}+\frac{1}{b})$ in the third column. Finally, Sg2im produces more unrecognisable symbols than ours, e.g. Sin and π in last column. In contrast, the symbols generated by our method present reasonable sizes and clear structures. Besides, symbols are reasonably placed in relation to each other, neither overlapping nor far apart. This is similar to real formulas written by humans. Such observations demonstrate the superiority of our method in generating both high-quality layouts and HME images.

Table 1. Comparison with existing methods on CHROME2019.

	SSIM ↑	FID ↓	WER ↓	ExpRate ↑
CycleGAN [71]	0.757	84.14	0.671	0.026
FormulaGAN [41]	0.724	74.68	0.601	0.066
Sg2im [24]	<u>0.787</u>	**10.02**	<u>0.393</u>	<u>0.219</u>
Ours	**0.793**	<u>10.98</u>	**0.326**	**0.316**

Quantitative Comparison. Table 1 shows the quantitative performance indices. Both Sg2im and our method achieves significant better indices than CycleGAN or FormulaGAN. This is consistent with the qualitative comparison results. According to SSIM and FID, Sg2im and our method almost perform equality in term of image quality. However, our method decreases the WER by 6.5 percent and improves the ExpRate by 10 percent, absolutely. Such distinct superiority in recognition implies that, the handwritten expressions generated by our method are signifi-

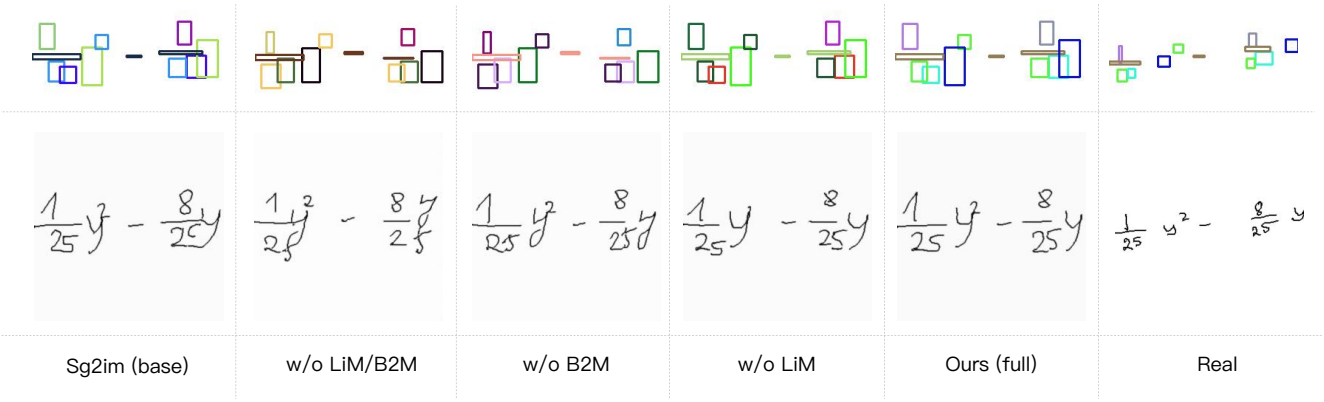

Figure 6. Illustration of the layouts and formula images generated by model variants, in the ablation study. The input formula is $\frac{1}{25}y^2 - \frac{8}{25}y$.

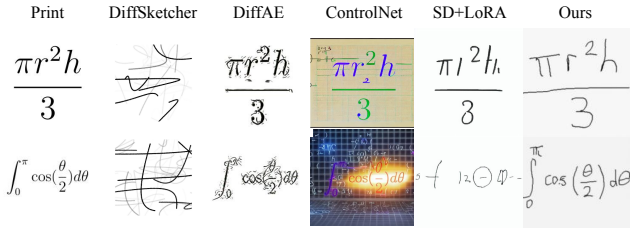

Figure 7. Comparison with diffusion models. DiffSketcher and DiffAE generate HMEs conditioned on print formulas; while ControleNet and SD+LoRA take both the print formula and a textual prompt ("*a handwritten mathematical expression of* `latex code`") as input. We use official diffusion models, and fine-tune DiffAE and SD+LoRA using HMEs.

cantly better than Sg2im, in terms of clarity and structure.

Comparison with Diffusion Models. We additionally compare with ControlNet [67], Stable Diffusion (SD) [39] + LoRA [20], DiffSketcher [56], and DiffAE [38]. As shown in Fig. 7, these diffusion models cannot generate high-quality HMEs with realistic handwritten styles or recognizable symbols. Besides, their computational complexity is much heavier than ours. Some recent methods, e.g. ChiroDiff [8] and SDT [7], can only generate handwritten images of one *single* character. Such comparison results demonstrate the advantage of our method in HMEG.

4.3. Ablation Study

We further analyse the impact of (i) the LiM training strategy, (ii) the layout discriminator (D_{box}), and (iii) the B2M module. To this end, we build model variants by:

- Using original training graphs, instead of sampled minimal graphs, to train the layout predictor (*w/o LiM*);
- Removing B2M from our full model, but using grids for embedding selection (*w/o B2M*);
- Removing both LiM and B2M from our full model (*w/o LiM/B2M*);
- Training the whole model following Sg2im, without using LiM, D_{box}, and B2M (*Sg2im (base)*).

We additionally report the mean Intersection over Union (mIoU) between the predicted layouts and the target ones.

Table 2. Performance indices w.r.t. the ablation study on CHROME19. The model variant in the last row is our baseline.

	SSIM ↑	FID↓	WER↓	ExpRate ↑	mIOU ↑
Ours (full)	**0.793**	10.98	**0.326**	**0.316**	**0.364**
w/o LiM	0.790	11.55	0.327	0.315	0.335
w/o B2M	0.790	11.46	0.332	0.308	0.354
w/o LiM/B2M	0.786	13.30	0.365	0.279	0.338
Sg2im (base)	0.787	**10.08**	0.393	0.219	0.324

As shown in Table 2, removing LiM or B2M slightly changes the indices about image quality, decreases the IoU. Specially, removing LiM decreases the mIoU by about 3 percent absolutely. This implies the significance of LiM in generating good layouts. In addition, if we removing D_{box} or B2M, along with LiM, the performance seriously decreases, almost in terms of all performance indices. This implies that our D_{box} and B2M are significant for boosting the quality of generated images from all aspects, including the fidelity (SSIM), the usability for recognition (WER & ExpRate), and the layout (mIoU). Fig. 6 illustrates the layouts and images generated by different model variants. All the model variants, except the full model, produce overlapped symbols or symbols in unnatural sizes. The image generated by our model presents no obvious defects and are easy to recognize. Such observations further demonstrate the significance of LiM, B2M, and layout discriminator in layout prediction and image generation.

4.4. Applications

Mathematical Expression Manipulation. In addition, we apply the previous learned model to expression manipulation using symbol graphs. Given a source symbol graph,

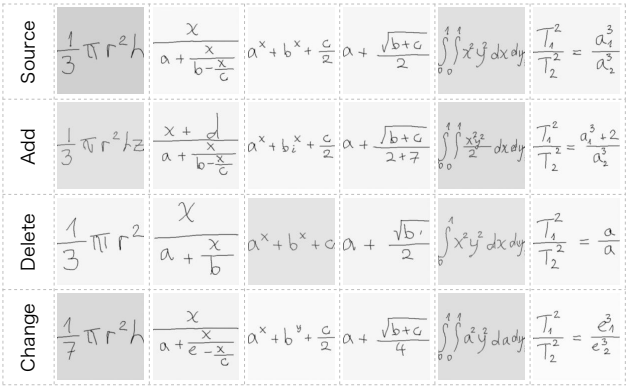

Figure 8. Illustration of expression manipulation using symbol graphs (LaTeX sequences).

we edit it by adding, deleting, or changing symbol nodes in it, and apply the previously learned generator to produce the corresponding HME image. As shown in Fig. 8, the generated image changes consistently with the editing operation. Besides, the manipulated images present natural layouts, recognizable symbols, and clear strokes.

Data Augmentation for HMER. Finally, we validate the possible application of HMEG in augmenting HMER models, since the generated data could improve the amount and diversity of training data. To this end, we conduct experiments on CROHME2014/2016/2019 [34], and use CAN [27] as the HMER model. Here, we train CAN with (1) the standard training data (denoted by CAN); or (2) with previous data augmentation, e.g. rotation, (denoted by CAN*); or (3) with our sythetic data augmentation, using 6,000 additional generated samples (denoted by CAN†); or (4) both previous and our data augmentation (denoted by CAN*†). The model is trained for 120 epochs under each setting, using the official training parameters. Afterward, we apply the learned HMER model to both generated and real data.

As shown in Table 3, our method achieves 5-11% absolute improvement over the baseline, and performs comparably to previous data augmentation, across all the datasets. Besides, our method can be *jointly* used with previous augmentation to further boost the performance (i.e. CAN*†). We additionally report the HMER performance, as well as the SSIM and StruRate, of our generated samples. The SSIM and StruRate values are high, and the ExpRate values (by CAN*/CAN†) approach those of real data, on all datasets. These results demonstrate the robustness of our method, in generating high-quality HMEs, with correct layouts and recognizable symbols.

Note: Previous HME synthesis methods [10, 25] synthesize HMEs by *shuffling* the symbols and layouts contained in original HME datasets. Neither of them is a *generative* method. And they may cause overfitting to existing datasets. In contrast, our pipeline allows excellent *flexibility* in generating *diverse* new samples. The experimental results also demonstrate the advantages of our approach in both HMEG and HMER.

Table 3. Results on CROHME14/16/19 testing sets and our generated images. *, †, *† denote using previous data augmentation, our synthetic data augmentation, and both, respectively.

	HMEG (*generated*)				**HMER** (*real*)			
	SSIM	CAN*	CAN†	StruRate	CAN	CAN*	CAN†	CAN*†
CROHME14	0.789	52.1	54.1	98.1	44.7	52.9	50.2	**55.4**
CROHME16	0.798	51.4	55.8	96.5	42.8	52.4	53.9	**57.6**
CROHME19	0.793	55.4	56.4	97.7	39.3	48.4	49.6	**58.5**

5. Conclusions

In this paper, we propose a novel method to solve the challenging task of generating HMEs from LaTeX sequences. We formulate it as a graph-to-image generation task, and focus on boosting the layout prediction without mask supervision. Experimental results demonstrate the effectiveness of our method, and its possible applications in expression manipulation and the challenging inverse task, i.e. HMER. Inspired by the remarkable process of generative augmentation in visual understanding [35], we believe that: the novel HMEG task we explore here has great potential to significantly boost HMER.

Due to computational limitations, we merely use naive networks in the implementation. The generated symbols are occasionally unrecognizable, especially in complex formulas. It's promising to boost the generation quality by using advanced networks (e.g. VQ-GAN [11] or diffusion models [15]). Besides, we will extend the proposed techniques to natural/artistic image generation by exploring the inspiring capacity of diffusion models, and other tasks of structured graph data. Finally, to further boost the HMER performance, it's promising to use the triples of {*LaTeX, predicted layout, generated image*} in HMEG as pseudo labeled data, or to use the layout discriminator for enhancing the layout detection module in an HMER pipeline.

Acknowledgements

We greatly appreciate Biao Ma's and Chang Jiang's help in the implementation of HMER. This work was supported in part by the National Natural Science Foundation of China under Grants U22A2096 and 61971172; in part by the Technology Innovation Leading Program of Shaanxi under Grant 2022QFY01-15; in part by the Fundamental Research Funds for the Central Universities under Grant QTZX23042 and ZYTS24012; and in part by the Proof of Concept Foundation of Xidian University Hangzhou Institute of Technology under Grant No. GNYZ2023YL0301.

References

[1] Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4561–4569, 2019. 3

[2] Xiaohang Bian, Bo Qin, Xiaozhe Xin, Jianwu Li, Xuefeng Su, and Yanfeng Wang. Handwritten mathematical expression recognition via attention aggregation based bi-directional mutual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 113–121, 2022. 2

[3] Stefan Bornholdt. Less is more in modeling large genetic networks. *Science*, 310(5747):449–451, 2005. 4

[4] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1511–1520, 2017. 3, 5

[5] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014. 2

[6] 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*, pages 5977–5986, 2023. 3

[7] 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*, pages 5977–5986, 2023. 7

[8] Ayan Das, Yongxin Yang, Timothy M Hospedales, Tao Xiang, and Yi-Zhe Song. Chirodiff: Modelling chirographic data with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. 7

[9] Helisa Dhamo, Azade Farshad, Iro Laina, Nassir Navab, Gregory D Hager, Federico Tombari, and Christian Rupprecht. Semantic image manipulation using scene graphs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5213–5222, 2020. 3, 5

[10] Lan-Fang Dong, Han-Chao Liu, and Xin-Ming Zhang. Synthetic data generation and shuffled multi-round training based offline handwritten mathematical expression recognition. *Journal of Computer Science and Technology*, 37(6): 1427–1443, 2022. 8

[11] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 8

[12] Wan-Cyuan Fan, Yen-Chun Chen, DongDong Chen, Yu Cheng, Lu Yuan, and Yu-Chiang Frank Wang. Frido: Feature pyramid diffusion for complex scene image synthesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 579–587, 2023. 1, 3

[13] Azade Farshad, Sabrina Musatian, Helisa Dhamo, and Nassir Navab. Migs: Meta image generation from scene graphs. *arXiv preprint arXiv:2110.11918*, 2021. 2, 3

[14] Yingnan Fu, Wenyuan Cai, Ming Gao, and Aoying Zhou. Symbol location-aware network for improving handwritten mathematical expression recognition. In *Proceedings of the 2023 ACM International Conference on Multimedia Retrieval*, pages 516–524, 2023. 1, 2

[15] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022. 8

[16] Roei Herzig, Amir Bar, Huijuan Xu, Gal Chechik, Trevor Darrell, and Amir Globerson. Learning canonical representations for scene graph to image generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pages 210–227. Springer, 2020. 3

[17] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, pages 6626–6637. 2017. 6

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[19] Hongyu Wang and Jianshu Zhang. Handwritten mathematical expression recognition (pytorch), 2019. https://github.com/whywhs/Pytorch-Handwritten-Mathematical-Expression-Recognition. 6

[20] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021. 7

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 1, 3, 4

[22] Maor Ivgi, Yaniv Benny, Avichai Ben-David, Jonathan Berant, and Lior Wolf. Scene graph to image generation with contextualized object layout refinement. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 2428–2432. IEEE, 2021. 2, 3

[23] Nikita Jaipuria, Xianling Zhang, Rohan Bhasin, Mayar Arafa, Punarjay Chakravarty, Shubham Shrivastava, Sagar Manglani, and Vidya N Murali. Deflating dataset bias using synthetic data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 772–773, 2020. 1

[24] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018. 1, 2, 3, 5, 6

[25] Vu Tran Minh Khuong, Ung Quang Huy, Nakagawa Masaki, and Minh Khanh Phan. Generating synthetic handwritten mathematical expressions from a latex sequence or a mathml script. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 922–927. IEEE, 2019. 8

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on*

Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. 5

[27] 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 *European Conference on Computer Vision*, pages 197–214. Springer, 2022. 1, 2, 8

[28] Wenbo Li, Pengchuan Zhang, Lei Zhang, Qiuyuan Huang, Xiaodong He, Siwei Lyu, and Jianfeng Gao. Object-driven text-to-image synthesis via adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12174–12182, 2019. 1, 3

[29] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015. 4

[30] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects. In *International conference on machine learning*, pages 3835–3845. PMLR, 2019. 2, 4

[31] Yikang Li, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[32] Zejian Li, Jingyu Wu, Immanuel Koh, Yongchuan Tang, and Lingyun Sun. Image synthesis from layout with locality-aware mask adaption. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13819–13828, 2021. 3

[33] Zhe Li, Xinyu Wang, Yuliang Liu, Lianwen Jin, Yichao Huang, and Kai Ding. Improving handwritten mathematical expression recognition via similar symbol distinguishing. *IEEE Transactions on Multimedia*, 2023. 2

[34] 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)*, pages 1533–1538. IEEE, 2019. 2, 5, 8

[35] Quang Nguyen, Truong Vu, Anh Tran, and Khoi Nguyen. Dataset diffusion: Diffusion-based synthetic data generation for pixel-level semantic segmentation. *Advances in Neural Information Processing Systems*, 36, 2024. 8

[36] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017. 5

[37] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019. 1

[38] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10619–10629, 2022. 7

[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1, 7

[40] Renato Sortino, Simone Palazzo, Francesco Rundo, and Concetto Spampinato. Transformer-based image generation from scene graphs. *Computer Vision and Image Understanding*, 233:103721, 2023. 3

[41] 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*, pages 46–54, 2021. 1, 2, 6

[42] Sitong Su, Lianli Gao, Junchen Zhu, Jie Shao, and Jingkuan Song. Fully functional image manipulation using scene graphs in a bounding-box free way. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1784–1792, 2021. 3

[43] Wei Sun and Tianfu Wu. Image synthesis from reconfigurable layout and style. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10531–10540, 2019. 3

[44] Wei Sun and Tianfu Wu. Learning layout and style reconfigurable gans for controllable image synthesis. *IEEE transactions on pattern analysis and machine intelligence*, 44(9): 5070–5087, 2021. 3

[45] Tristan Sylvain, Pengchuan Zhang, Yoshua Bengio, R Devon Hjelm, and Shikhar Sharma. Object-centric image generation from layouts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2647–2655, 2021. 3

[46] Thanh-Nghia Truong, Cuong Tuan Nguyen, and Masaki Nakagawa. Syntactic data generation for handwritten mathematical expression recognition. *Pattern Recognition Letters*, 153:83–91, 2022. 1, 2

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Llion Jones, Jakob Uszkoreit, Aidan N Gomez, and Lukasz Kaiser. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008. 2017. 2

[48] Chi Wang, Min Zhou, Tiezheng Ge, Yuning Jiang, Hujun Bao, and Weiwei Xu. Cf-font: Content fusion for few-shot font generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1858–1867, 2023. 3

[49] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Process*, 13(4):600–612, 2004. 6

[50] Chuan Wen, Yujie Pan, Jie Chang, Ya Zhang, Siheng Chen, Yanfeng Wang, Mei Han, and Qi Tian. Handwritten chinese font generation with collaborative stroke refinement. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3882–3891, 2021. 3

[51] Chenxi Whitehouse, Monojit Choudhury, and Alham Fikri Aji. Llm-powered data augmentation for enhanced crosslingual performance. *arXiv preprint arXiv:2305.14288*, 2023. 1

[52] Changjie Wu, Jun Du, Yunqing Li, Jianshu Zhang, Chen Yang, Bo Ren, and Yiqing Hu. Tdv2: A novel tree-structured decoder for offline mathematical expression recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2694–2702, 2022. 2

[53] Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. Handwritten mathematical expression recognition via paired adversarial learning. *International Journal of Computer Vision*, 128:2386–2401, 2020. 2

[54] Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. Graph-to-graph: towards accurate and interpretable online handwritten mathematical expression recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2925–2933, 2021. 2

[55] Yang Wu, Pengxu Wei, and Liang Lin. Scene graph to image synthesis via knowledge consensus. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2856–2865, 2023. 3

[56] Ximing Xing, Chuang Wang, Haitao Zhou, Jing Zhang, Qian Yu, and Dong Xu. Diffsketcher: Text guided vector sketch synthesis through latent diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 7

[57] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324, 2018. 1

[58] Yuan Xue, Yuan-Chen Guo, Han Zhang, Tao Xu, Song-Hai Zhang, and Xiaolei Huang. Deep image synthesis from intuitive user input: A review and perspectives. *Computational Visual Media*, 8:3–31, 2022. 1

[59] Chen Yang, Jun Du, Jianshu Zhang, Changjie Wu, Mingjun Chen, and JiaJia Wu. Tree-based data augmentation and mutual learning for offline handwritten mathematical expression recognition. *Pattern Recognition*, 132:108910, 2022. 1, 2

[60] Ling Yang, Zhilin Huang, Yang Song, Shenda Hong, Guohao Li, Wentao Zhang, Bin Cui, Bernard Ghanem, and Ming-Hsuan Yang. Diffusion-based scene graph to image generation with masked contrastive pre-training. *arXiv preprint arXiv:2211.11138*, 2022. 1, 3

[61] Jun Yu, Xingxin Xu, Fei Gao, Shengjie Shi, Meng Wang, Dacheng Tao, and Qingming Huang. Toward realistic face photo–sketch synthesis via composition-aided gans. *IEEE transactions on cybernetics*, 51(9):4350–4362, 2020. 1

[62] 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*, pages 4553–4562, 2022. 1

[63] Richard Zanibbi and Dorothea Blostein. Recognition and retrieval of mathematical expressions. *International Journal on Document Analysis and Recognition (IJDAR)*, 15:331–357, 2012. 2, 3

[64] Jinshan Zeng, Qi Chen, Yunxin Liu, Mingwen Wang, and Yuan Yao. Strokegan: Reducing mode collapse in chinese font generation via stroke encoding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3270–3277, 2021. 3

[65] Yu Zeng, Zhe Lin, Jianming Zhang, Qing Liu, John Collomosse, Jason Kuen, and Vishal M Patel. Scenecomposer: Any-level semantic image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22468–22478, 2023. 1

[66] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *IEEE TPAMI*, PP(99), 2017. 1

[67] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 1, 7

[68] Yaping Zhang, Shuai Nie, Wenju Liu, Xing Xu, Dongxiang Zhang, and Heng Tao Shen. Sequence-to-sequence domain adaptation network for robust text image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[69] Bo Zhao, Weidong Yin, Lili Meng, and Leonid Sigal. Layout2image: Image generation from layout. *International Journal of Computer Vision*, 128:2418–2435, 2020. 1, 3

[70] Wenqi Zhao and Liangcai Gao. Comer: Modeling coverage for transformer-based handwritten mathematical expression recognition. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 392–408. Springer, 2022. 2

[71] J. Zhu, T. Park, P. Isola, and A. A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2242–2251, 2017. 6

[72] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Sean: Image synthesis with semantic region-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5104–5113, 2020. 1