

# Translating Math Formula Images To LaTeX Sequences

## Abstract

Recognizing mathematical formulas in images and translating them into LaTeX sequences, both printed and handwritten, is challenging due to the complexity of two-dimensional formulas and lack of training data. Traditional methods can only handle simple formulas and are not effective for complex formulas. This report introduces the Sumen (**Scaling Up Image-to-LaTeX Performance**) model, an encoder-decoder architecture based on Transformer with attention mechanism trained on the largest dataset from previous works. The model achieves a BLEU score of 95.59, Edit Distance (ED) of 97.3, and Exact Match (EM) of 69.23 on the img2latex100k benchmark. On the CROHME 2014/2016/2019 benchmark, the corresponding results on Expression Recognition Rates (ExpRate) are 58.01/82.39/78.99 and Word Error Rate (WER) are 9.46/2.55/4.51. All of metrics outperform state-of-the-art methods on both printed and handwritten formulas.

# Contents

<b>Abstract</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Overview . . . . .	3
1.1.1 Context . . . . .	3
1.1.2 Math Formula Images To Latex . . . . .	3
1.2 Background . . . . .	5
1.3 Specific Works . . . . .	5
<b>2 Related Works</b>	<b>6</b>
<b>3 Materials and Methods</b>	<b>8</b>
3.1 Dataset . . . . .	8
3.2 Methods . . . . .	12
3.2.1 Encoder . . . . .	13
3.2.2 Decoder . . . . .	14
3.2.3 Data Augmentation . . . . .	14
3.3 Implementation Details . . . . .	15
<b>4 Results</b>	<b>16</b>
4.1 Metrics . . . . .	16
4.2 Printed Mathematical Expressions Benchmark . . . . .	16
4.3 Handwritten Mathematical Expressions Benchmark . . . . .	17
<b>5 Conclusions</b>	<b>20</b>
<b>References</b>	<b>21</b>

# 1 Introduction

## 1.1 Overview

### 1.1.1 Context

Mathematical formulas are essential for life and are widely used in various fields. They are used to describe algorithms, illustrate ideas, clarify complex concepts, and create clear and consistent content. To express mathematical formulas in documents, we need to use markup languages. LaTeX is a language used for typesetting and formatting mathematical formulas professionally. Its application is spreading from science to technology, from papers to websites. In the scientific field, LaTeX is widely used to write articles, scientific research reports, blogs, and books. It can show the intuition of formulas and is a powerful tool to represent them clearly, accurately, and easily readable. In addition, in the field of education, LaTeX is often used to write mathematical formulas in lectures, teaching materials, and tests. Mathematical formulas can be used to explain complex concepts more clearly and attractively to students.

Representing mathematical formulas with LaTeX is a real challenge, especially for non-professionals. Even for experts, typing long and complex mathematical formulas is not easy. Mathematical expressions that are too long or too complex can make users easily make some minor mistakes, which can lead to time loss and inconvenience in editing. And these errors happen frequently. If a formula is too long, repairing or adjusting it can be very complicated, we may need to scan each part of the formula to be able to fix the error, which can make us confused and impatient when we have to fix each character. And for those who are new to LaTeX, typing each complex code can be a big challenge, requiring a lot of time and patience.

### 1.1.2 Math Formula Images To Latex

To help people apply LaTeX to write mathematical formulas easily and quickly, a model has been developed to support converting mathematical formula images into corresponding LaTeX sequences. This way, users can easily convert handwritten mathematical formulas on paper and convert them into LaTeX format, saving time and effort compared to manual typing. In addition, our model can also be used to support users in practicing how to use LaTeX, helping users to easily understand the structure of writing

LaTeX code and learn to use it faster. This is especially useful for beginners with LaTeX, as they can practice writing mathematical formulas and see the results immediately through our model. It helps users learn effectively in learning LaTeX, thereby promoting the learning process and approaching this language flexibly and efficiently.

Converting math formula images to LaTeX, also known as img2latex, is the process of translating mathematical formulas in images into LaTeX sequences. It was created to help scientists, teachers, and non-professionals easily convert mathematical formulas from images into LaTeX code and display them in their documents. Formula recognition has two problems: recognizing images containing handwritten mathematical expressions or printed mathematical expressions. Handwritten formula recognition is more difficult than printed formula recognition due to many reasons such as lack of large enough data, confusion and lack of clarity due to bad handwriting.

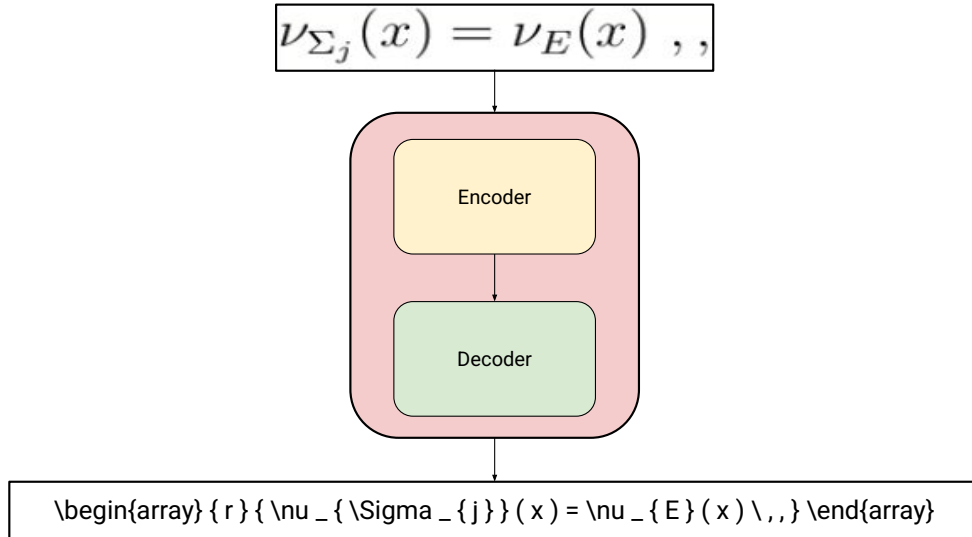


Figure 1: Image-to-latex workflow

An end-to-end approach means that the model receives image input and directly generates a LaTeX sequences without going through the intermediate steps illustrated in Figure 1. Specifically, the model takes raw image input without the need for an OCR engine, bypassing the need for an OCR engine to help prevent images from being segmented.

## 1.2 Background

Previously, traditional methods such as the INFITY system [33] created rules based on structural analysis methods and syntactic parsers. These methods could solve simple mathematical formulas such as superscript, subscript, special symbols, and fractions. However, they could not solve complex and long formulas because they only recognized characters and did not understand the relationship between characters and multi-line formulas.

In recent years, the advent of deep neural networks has led to the proposal of methods based on the encoder-decoder architecture. These methods are widely used and have achieved good results in many areas such as image captioning, machine translation, text to image, and speech to text. The encoder-decoder architecture combines computer vision and natural language processing. The encoder is responsible for processing the image and capturing important information features. The decoder then receives these feature information and generates the corresponding LaTeX string for the formulas in the image. The encoder-decoder architecture simplifies in the process and has achieved many successes in this field. The model receives the input and directly generates the output without the need to complicate the data, such as segmenting the location of text segments in the image.

Although the encoder-decoder architecture has been working well in recent years, previous methods have only focused on the Printed Mathematical Expression Recognition (PMER) [12, 13, 31, 37, 38, 47] or Handwritten Mathematical Expression Recognition (HMER) [45, 8, 46, 19, 10, 41, 43]. However, they have not been very good at both tasks in the same method.

## 1.3 Specific Works

The encoder-decoder architecture is utilized and continuously improved. A large dataset of approximately 3.4 million image-text pairs is proposed to achieve robust generalization. Both handwritten and printed mathematical expression data are trained simultaneously, enabling the model to share LaTeX representation knowledge. Handwritten mathematical expressions are treated as a form of data augmentation. The model supports converting images to LaTeX for both printed and handwritten formulas, achieving strong results on various benchmarks.

## 2 Related Works

The advent of deep neural networks has replaced classical methods with encoder-decoder architectures and brought about good results and real success in the field of Computer Vision & Natural Language Processing. The encoder uses models such as Convolutional Neural Networks (CNN) and Vision Transformer (ViT) [15]. CNN is one of the most popular models used in image processing, with the ability to automatically extract features from image inputs through convolution and pooling layers, CNN has proven its superiority in object recognition and image classification. Some popular CNNs today include EfficientNet [34], which is designed to optimize computational efficiency and accuracy by increasing the model size in depth, width, and resolution. DenseNet [17] is a densely connected CNN architecture where each layer is connected to every other layer in a dense manner. This enhances information flow and addresses the vanishing gradient problem. Meanwhile, ViT is a neural network architecture that uses Transformer, originally developed for natural language processing, but has been applied to the field of computer vision. In ViT network, images are divided into patches and represented as vectors, similar to how Transformer represents words in NLP. Each vector represents a part of the image and is fed into the Transformer network to extract information and perform tasks such as classification or object detection. ViT has demonstrated the power of Transformer in computer vision, which is no less than that of Transformer in natural language processing.

Decoder based on Recurrent Neural Network (RNN) has the disadvantages of vanishing gradient and exploding gradient, making it challenging to work with long sequences. It is also slow to compute and can only learn short-term memory [28]. For complex mathematical formula markup chains that can exceed hundreds of LaTeX tokens, the hidden state vector in RNN is not enough to compress all the information from the encoder. Long Short Term Memory (LSTM) was introduced to solve the problems of RNN and to scale up to capture long-term memory when needed. It mitigates the vanishing gradient problem and can forget irrelevant information through the forget gate. However, LSTM is complex, requires significant time and computational resources, and cannot be parallelized. Transformer [36] was introduced based on the self-attention mechanism and has gradually replaced methods like RNN and LSTM due to its convenient parallel processing and ability to overcome the limitations mentioned above. It is widely used with Large Language

Models (LLMs) and has achieved state-of-the-art results.

Methods using encoder-decoder architecture such as [13] propose neural encoder-decode with coarse-to-fine attention mechanism, the author uses CNN encoder to extract feature in the image and RNN decoder implements a conditional language model over the vocabulary. [31] introduces a neural transducer model with visual attention, which uses CNN as encoder and RNN as decoder combined with using beam search in the inference process. [38] proposed a method including CNN combined with positional encoding used in encoder to extract features, features will be augmented with 2D positional encoding before being unfolded into a vector and fed into the LSTM decoder to translate into a sequence of LaTeX tokens. [47] has proposed a model applying Transformer-based encoder-decoder architecture, in which the encoder uses ViT and takes the idea of machine translation applied to img2latex task, in addition this method combines using the YOLO model [30] for the preprocessing step to separate single-line formulas from multi-line formulas to improve the model’s accuracy.

Autoregressive-decoder methods like [43, 42] idea of learning and generating text strings in left-to-right direction. Since then, the BTTR [46] and ABM [8] methods were born and introduced the novel bidirectional training strategy with the purpose of being able to learn LaTeX sequences in left-to-right and right-to-left directions on the Transformer decoder, helping the model learn well and achieve higher accuracy. But entails parameters and longer training time. Inspired by the coverage mechanism in RNN, CoMER [45] proposes a model to improve the Transformer’s shortcomings in terms of sufferings from the lack of coverage problem, using Attention Refinement Module (ARM) to fine-tune attention weights with past alignment information without hurting its parallelism and performs better than vanilla transformer decoder and RNN decoder in the HMER task.

## 3 Materials and Methods

### 3.1 Dataset

The formula recognition task is divided into two domains: handwritten and printed mathematical formulas. Different datasets are utilized to train and evaluate the model based on these two domains. The largest dataset to date has been collected and built from online sources, creating a robust and well-generalizable dataset. This dataset consists of approximately 3.4 million image-text pairs, including 200,330 samples of handwritten mathematical expressions and 3,237,250 samples of printed mathematical expressions. The percentage of data is illustrated in Figure 2 and data details are shown in Figure 3. The dataset is published at [16].

**Printed mathematical expressions:** The data is collected from the Im2latex-100k dataset [12], I2L-140K Normalized dataset and Im2latex-90k Normalized dataset [31], Im2latex-170k dataset [2], Im2latex-230k dataset [3], latex-formulas dataset [7] and Im2latex dataset [4].

**Handwritten mathematical expressions:** The data is collected from the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) dataset [25, 26, 24], Aida Calculus Math Handwriting Recognition Dataset [1] and Handwritten Mathematical Expression Convert LaTeX [5]

**Pre-processing:** Due to the large dataset and the fact that the same mathematical formula can be represented in different LaTeX string formats in an image, it is easy to cause polymorphic ambiguity. To address this issue, the normalization method with the KaTeX parser [6] is employed. The raw LaTeX strings are converted into an abstract syntax tree, followed by the application of safe normalizing tree transformations to eliminate ambiguity in the LaTeX markup strings. This method has been used in previous works, including [12, 38].



Items	Amount of data (train)
Im2latex-100k dataset [12]	93,925 image-text pairs
I2L-140K Normalized dataset and Im2latex-90k Normalized dataset [31]	132,500 image-text pairs
Im2latex-170k dataset [2]	165,477 image-text pairs
Im2latex-230k dataset [3]	234,311 image-text pairs
latex-formulas dataset [7]	1,035,945 image-text pairs
Im2latex dataset [4]	1,586,584 image-text pairs
CROHME [25, 26, 24]	10,968 image-text pairs
Handwritten Mathematical Expression Convert LaTeX [5]	11,181 image-text pairs
Aida Calculus Math Handwriting Recognition Dataset [1]	100,000 image-text pairs

Table 1: Source of dataset

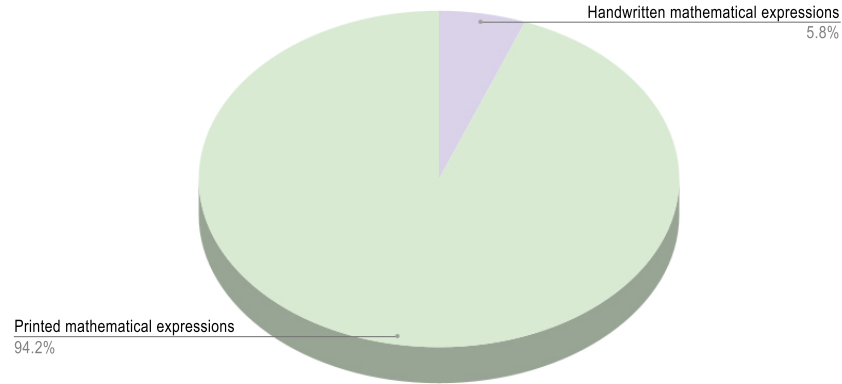


Figure 2: Overall dataset

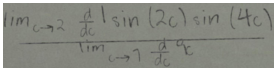
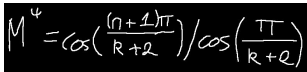
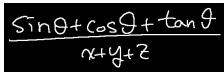
Images	Labels
$y = \begin{pmatrix} 0 & q_1 & 0 & 0 \\ 0 & 0 & q_2 & 0 \\ 0 & 0 & 0 & q_3 \\ q_4 & 0 & 0 & 0 \end{pmatrix}$	$y = \left( \begin{array}{cccc} c & c & c & c \\ \{0\} & \{q_{-1}\} & \{0\} & \{0\} \\ \{q_{-2}\} & \{0\} & \{0\} & \{0\} \\ \{0\} & \{0\} & \{q_{-3}\} & \{0\} \end{array} \right)$
$G_N = g_s^2 \left( \frac{E}{E_s} \right)^{D-2} = \left( \frac{\lambda}{N} \right)^2 \left( \frac{E}{E_s} \right)^{D-2-2\alpha}$	$G_{-N} = g_{-s}^2 \left( \frac{E}{E_s} \right)^{D-2} = \left( \frac{\lambda}{N} \right)^2 \left( \frac{E}{E_s} \right)^{D-2-2\alpha}$
$ n\rangle\langle n+\ell  = 2(-1)^n \sqrt{\frac{n!}{(n+\ell)!}} \left( \frac{2r^2}{\theta} \right)^{\ell/2} L_n^\ell(2r^2/\theta) e^{-r^2/\theta} e^{i\ell\varphi},$	$ n\rangle\langle n+\ell  = 2(-1)^n \sqrt{\frac{n!}{(n+\ell)!}} \left( \frac{2r^2}{\theta} \right)^{\ell/2} L_n^\ell(2r^2/\theta) e^{-r^2/\theta} e^{i\ell\varphi},$
$\phi(\xi) = V(t,x), \xi = x - ct, c > 0$	$\phi(\xi) = V(t,x), \xi = x - ct, c > 0$
$\mathbb{E}\left[\int_0^T  Z^\epsilon(s) ^2 ds\right] \leq C_2(1+ x ^2),$	$\mathbb{E}\left[\int_0^T  Z^\epsilon(s) ^2 ds\right] \leq C_2(1+ x ^2),$
$U_S - S U = \begin{pmatrix} 0 & 0 & 0 & 0 \\ a-f & m & c-g & m \\ 0 & 0 & 0 & 0 \\ -a+f & -m & -c+g & -m \end{pmatrix}$	$U_S - S U = \begin{pmatrix} 0 & 0 & 0 & 0 \\ a-f & m & c-g & m \\ 0 & 0 & 0 & 0 \\ -a+f & -m & -c+g & -m \end{pmatrix}$
$\Delta_j(\alpha) = F_j^{-1}(\Delta^{(0)}(\alpha)F_j) \quad (\alpha \in U(s(2))).$	$\Delta_j(\alpha) = F_j^{-1}(\Delta^{(0)}(\alpha)F_j) \quad (\alpha \in U(s(2))).$
	$\lim_{c \rightarrow 2} \frac{d}{dx} \sin(2c) \sin(4c) = \lim_{c \rightarrow 7} \frac{d}{dx} 9c$
$\int_{\sqrt{\Omega}}^{S_0(X)} dS_0 \sqrt{\Omega + S_0^2} = \int_{\mu_0}^X dt \sqrt{\Lambda + \varphi^2(X)}$	$\int_{\sqrt{\Omega}}^{S_0(X)} dS_0 \sqrt{\Omega + S_0^2} = \int_{\mu_0}^X dt \sqrt{\Lambda + \varphi^2(X)}$
	$M^\psi = \cos(\frac{(n+1)\pi}{R+2}) / \cos(\frac{\pi}{R+2})$
	$\frac{\sin\theta + \cos\theta + \tan\theta}{x+y+z}$
$\frac{\partial_\Delta f}{\partial t} = \frac{f(r+\Delta, t) - f(r, t)}{\Delta}; \quad \frac{\partial_\Delta f}{\partial t} = \frac{f(r, t+\Delta) - f(r, t)}{\Delta}.$	$\frac{\partial_\Delta f}{\partial t} = \frac{f(r+\Delta, t) - f(r, t)}{\Delta}; \quad \frac{\partial_\Delta f}{\partial t} = \frac{f(r, t+\Delta) - f(r, t)}{\Delta}.$

Figure 3: Data examples

**Data Format:** The dataset was collected from multiple sources, resulting in various formats. Therefore, processing and storing the data in a common format was necessary. The dataset consists of four folders: train, test, val, and root, as shown in Figure 4. The root folder contains images in gray, png, jpg, and bmp formats. For the train, test, and val folders, we store the image paths and corresponding LaTeX codes in CSV (Comma-Separated Values) files with two columns. CSV is widely used in AI/ML and Data Science. It is stored in a simple tabular format, which is flexible and widely supported. In the train and val folders, there are files such as handwritten mathematical expressions and total for both datasets.

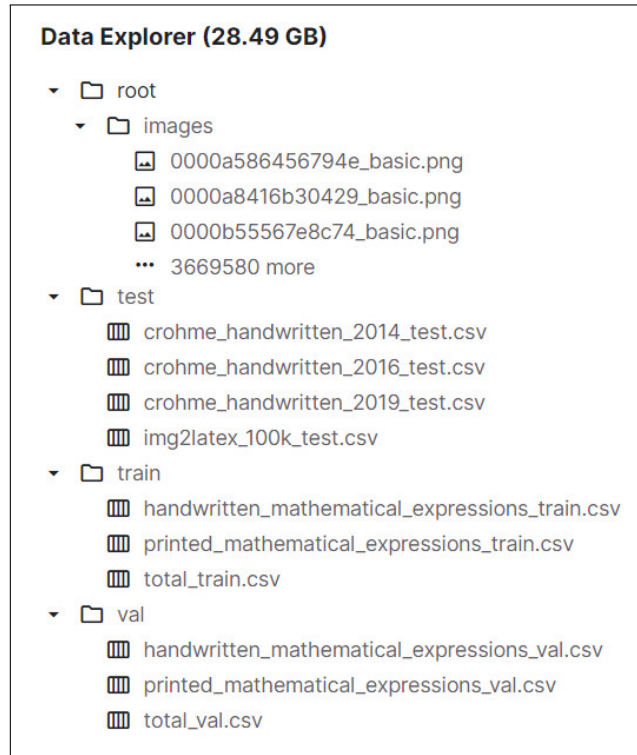


Figure 4: Data format

## 3.2 Methods

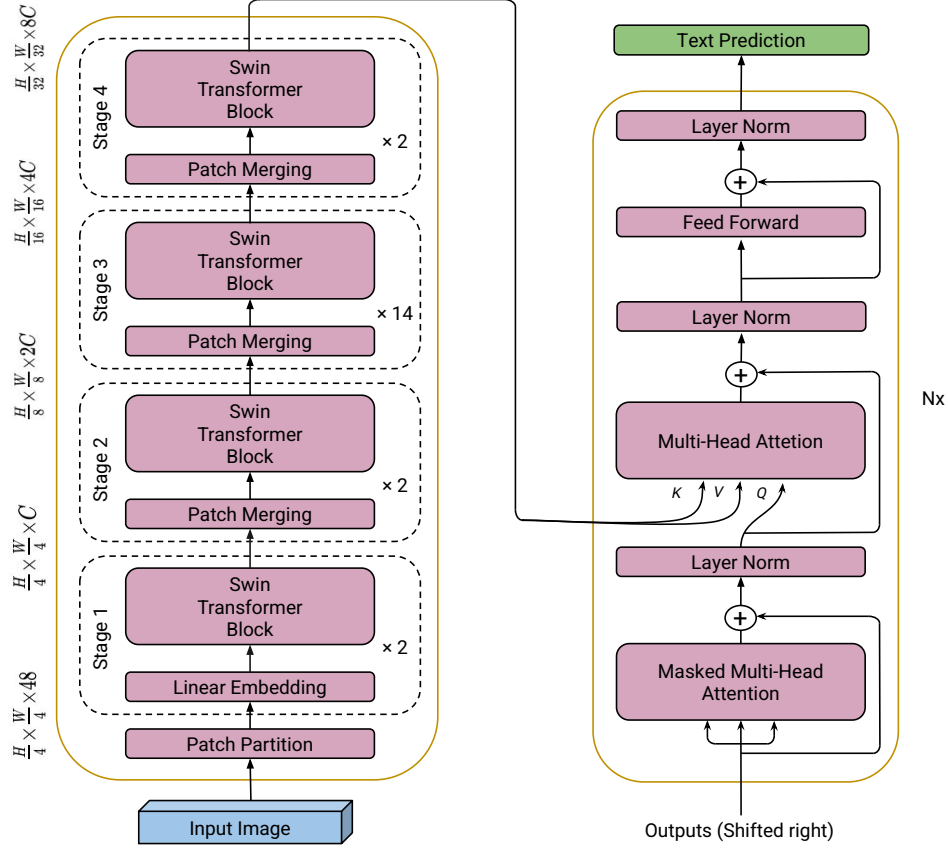


Figure 5: Overview of Sumen architecture (left: encoder, right: decoder)

The architecture we use in this project is Nougat [9], which is a transformer-based encoder-decoder architecture. Nougat is a Visual Transformer model that performs an Optical Character Recognition (OCR) task to understand scientific documents and translate them into markup language. The encoder is Swin Transformer [23] and the decoder is decoder-Transformer [36]. The entire architecture of the model is shown in Figure 5. The weights are initialized from the Nougat model.

### 3.2.1 Encoder

The advent of Swin Transformer is a breakthrough in the field of computer vision, combining the power of the attention mechanism and the hierarchical architecture of CNN. In which width and height are reduced and channels are increased in the later layers, providing flexibility to scale different image sizes. Some notable features are shown in Figure 6 as follows: Window-based self-attention is used to attend to patches or tokens in a window (local attention). However, sharing information across different windows is crucial to understanding the relationships between objects in the image. Therefore, shifted windows allow windows to communicate, which is an idea based on stride in CNN but with a different variation called cyclic shift. The model takes an input of an image with size  $H \times W \times 3$ , which is then used to extract important information from the image and output the encoder's  $KV$  feature vectors.

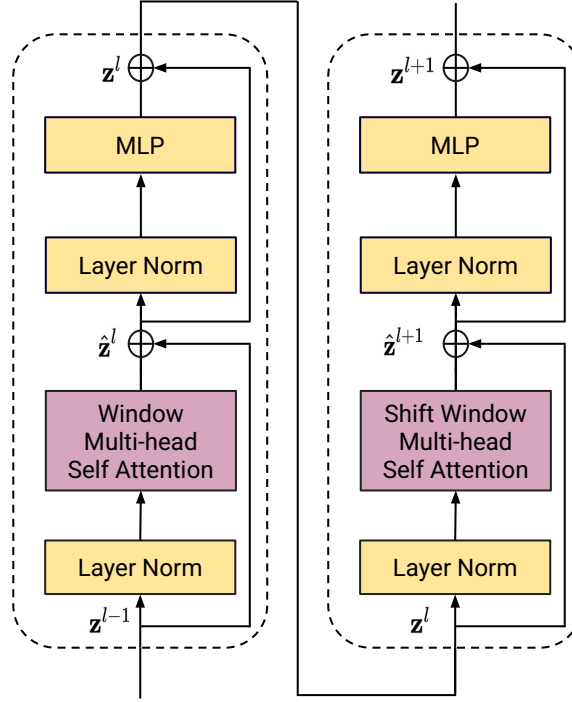


Figure 6: Swin transformer block

### 3.2.2 Decoder

The decoder uses decoder-Transformer, which is an auto-regressive language modeling (unidirectional), meaning that the model reads words only in the left-to-right direction. The decoder receives important information in the image represented by feature vectors  $KV$  from the encoder through cross multi-head attention and generates LaTeX sequences corresponding to the formulas in the image.

### 3.2.3 Data Augmentation

Image augmentation methods similar to those used in Nougat are applied during training, introducing random changes such as RGB shift, bitmap, erosion, dilation, shift-scale rotation, grid distortion, affine transformations, elastic transform, random brightness contrast, image compression, Gaussian noise, and Gaussian blur. These augmentations increase dataset variation, creating a more diverse dataset and improving the model’s generalization ability, an example is illustrated in Figure 7.

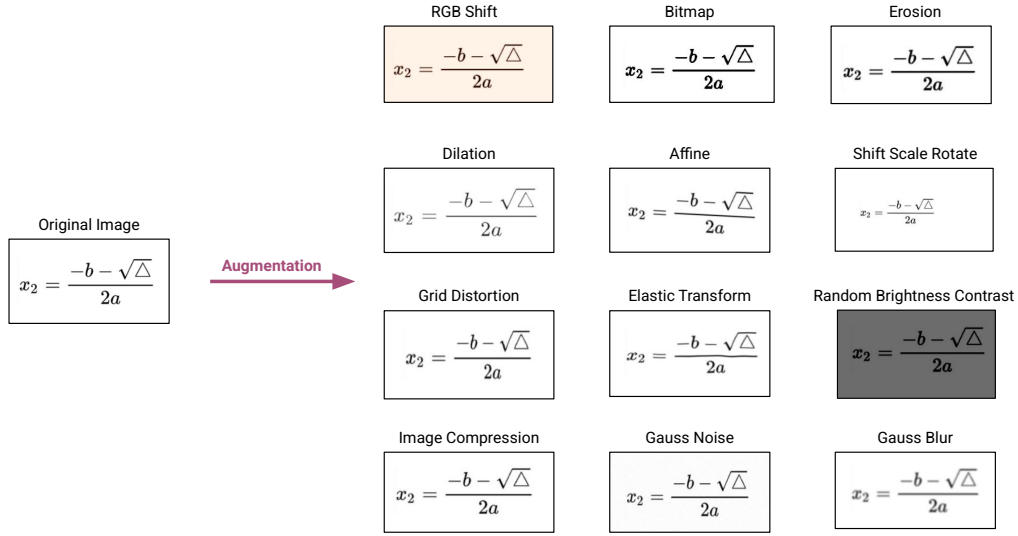


Figure 7: Some examples of transformations of the original image that will create a new training sample.

### 3.3 Implementation Details

The entire model is used using Pytorch & Transformer framework, During training, the Lion optimizer [11] is utilized with weight decay = 1e-2,  $\beta_1 = 0.95$  and  $\beta_2 = 0.98$ . A maximum length of 512 and an image resolution of  $224 \times 768$  (height  $\times$  width), learning rate of 1e-4 with 1k warmup steps based on cosine schedule. The entire model was trained in 10 epochs with batch size of 2048 with gradient accumulation.

## 4 Results

### 4.1 Metrics

The following metrics are used for the test dataset:

- **Bilingual Evaluation Understudy (BLEU)**: Compares the similarity between the machine translation and the reference translation. BLEU uses n-grams (combinations of n words) to measure the level of overlap between text strings [27].
- **Edit Distance (ED)**: Or Levenshtein distance [21], measures the minimum number of edit operations required to transform one text string into another. Edit operations include insertion, deletion, and substitution of characters. The result is calculated by dividing the total number of operations performed by the total number of words in the reference. To compare with other methods, we take  $1 - \text{minimum ED}$ .
- **Exact Match (EM)**: EM checks whether the machine translation or model outputs match the references exactly (100%). If it matches the references, the result is calculated by dividing the total number of correct outputs by the total number of references.
- **Word Error Rate (WER)**: WER uses edit distance to calculate the accuracy percentage. Instead of using the distance between phonemes, it uses the distance between words. The final result is calculated by dividing the total number of substitutions + deletions + insertions by the total number of words in the reference.
- **Expression Recognition Rates (ExpRate)**: ExpRate uses edit distance to calculate the accuracy of the model. For each line with distance = 0, the total number of correct lines is increased by 1. The final result is the number of correct lines divided by the total number of lines.

### 4.2 Printed Mathematical Expressions Benchmark

The Im2latex-100k dataset is used for experiments on the PMER task. This dataset contains approximately 100,000 real-world mathematical expressions rendered from public papers on the arxiv.org server and is widely recognized



for the printed formula recognition task. The test set consists of 10,285 samples. Evaluation metrics such as BLEU score (4-gram), Exact Match (EM) score, and Edit Distance (ED) score are employed for assessment on this dataset. The results shown in the Table 2 indicate that the model performs very well overall in terms of the entire string sequence on BLEU and ED, with the highest result, while EM is only at an average level and not the highest. This issue arises when one or more characters are incorrect, often resulting in zero matches for the entire string.

Model	BLEU	Edit Distance	Exact Match
INFTY [33]	66.65	96.4	59.6
CNNENC [12]	75.01	61.17	53.53
WYGIWYS [12]	87.73	87.60	77.46
MI2LS w/o Reinforce [38]	89.08	91.09	79.39
MI2LS with Reinforce [38]	90.28	92.28	82.33
DoubleAttention [38]	88.42	88.57	79.81
DenseNet [37]	88.25	91.57	-
MathBERT [29]	90.45	90.11	<b>87.52</b>
Zhou <i>et al.</i> [47]	92.11	90.0	60.2
Wang <i>et al.</i> [37]	85.71	90.25	28.68
DenseNet(2 blocks)	85.82	91.38	35.68
C-S attention	86.54	90.75	31.79
DenseNet + C-S	88.25	91.57	37.09
Sumen-base (*)	<b>95.59</b>	<b>97.3</b>	69.23

Table 2: Comparison of performance printed formula recognition task with previous methods on Im2latex-100k test set.

### 4.3 Handwritten Mathematical Expressions Benchmark

The CROHME dataset is used to demonstrate the effectiveness of the model on the HMER task. This is a large open dataset for handwritten mathematical expressions. The test set includes three versions: CROHME 2014 (986 samples), CROHME 2016 (1,147 samples), and CROHME 2019 (1,199 samples). Expression-level metrics are selected for evaluation: ExpRate (%),  $\text{ExpRate} \leq 1$  error (%),  $\text{ExpRate} \leq 2$  error (%), and  $\text{ExpRate} \leq 3$  error (%) provided by the CROHME 2019 organizers [24]. Furthermore, the Word

Error Rate (WER) metric is used to evaluate errors at the word level. The results shown in the Table 4 & 3, indicate that the model achieves the best results on 4 out of 4 ExpRate metrics for CROHME 16&19. However, on CROHME 14, the best result is achieved on only 1 out of 4 ExpRate metrics. Overall, the model outperforms other models.

Dataset	Model	ExpRate	WER
CROHME 14	Dense [43]	50.1	13.9
	Dense+MSA [43]	52.8	12.9
	ABM [43]	56.85	10.1
	Sumen-base (*)	<b>58.01</b>	<b>9.46</b>
CROHME 16	Dense [43]	47.5	15.4
	Dense+MSA [43]	50.1	13.7
	Sumen-base (*)	<b>82.39</b>	<b>2.55</b>
CROHME 19	Sumen-base (*)	<b>78.99</b>	<b>4.51</b>

Table 3: Comparison of performance handwritten formula recognition task with previous methods on CROHME 2014/2016/2019 test sets based on word error rate.

Dataset	Model	ExpRate	$\leq 1$ error	$\leq 2$ error	$\leq 3$ error
CROHME 14	DenseWAP [43]	43.0	57.8	61.9	-
	DenseWAP-TD [44]	49.1	64.2	67.8	-
	WS-WAP [35]	53.65	-	-	-
	Li <i>et al.</i> [22]	56.59	69.07	75.25	78.60
	Ding <i>et al.</i> [14]	58.72	-	-	-
	BTTR [46]	53.96	66.02	70.28	-
	BTTR (CoMER) [45]	55.17	67.85	72.11	74.14
	CoMER [45]	<b>59.33</b>	71.70	75.66	77.89
	PAL [39]	39.66	56.80	68.51	-
	WAP [42]	46.55	61.16	65.21	66.13
	PGS [20]	48.78	66.13	73.94	-
	PGS-v2 [40]	48.88	64.50	69.78	-
	DLA [18]	49.85	-	-	-
	ABM [8]	56.85	<b>73.73</b>	<b>81.24</b>	-
	WYGIWYS [12]	36.4	-	-	-
	Sumen-base (*)	58.01	72.11	80.22	<b>85.04</b>
CROHME 16	DenseWAP [43]	40.1	54.3	57.8	-
	DenseWAP-TD [44]	48.5	62.3	65.3	-
	WS-WAP [35]	51.96	64.34	70.10	72.97
	Li <i>et al.</i> [22]	54.58	69.31	73.76	76.02
	Ding <i>et al.</i> [14]	57.72	70.01	76.37	78.90
	BTTR [46]	52.31	63.90	68.61	-
	BTTR (CoMER) [45]	56.58	68.88	74.19	76.90
	CoMER [45]	59.81	74.37	80.30	82.56
	WAP [42]	44.55	57.10	61.55	62.34
	PGS [20]	36.27	-	-	-
	PGS-v2 [40]	49.61	64.08	70.27	-
	DLA [18]	47.34	-	-	-
	ABM [8]	52.92	69.66	78.73	-
	Sumen-base (*)	<b>82.39</b>	<b>89.97</b>	<b>94.42</b>	<b>95.99</b>
CROHME 19	DenseWAP [43]	41.7	55.5	59.3	-
	DenseWAP-TD [44]	51.4	66.1	69.1	-
	Ding <i>et al.</i> [14]	61.38	75.15	80.23	82.65
	BTTR [46]	52.96	65.97	69.14	-
	BTTR (CoMER) [45]	59.55	72.23	76.06	78.40
	CoMER [45]	62.97	77.40	81.40	83.07
	ABM [8]	53.96	71.06	78.65	-
	Sumen-base (*)	<b>78.99</b>	<b>86.22</b>	<b>90.5</b>	<b>92.07</b>

Table 4: Comparison of performance handwritten formula recognition task with previous methods on CROHME 2014/2016/2019 test sets based on expression recognition rates score.

## 5 Conclusions

A transformer-based encoder-decoder architecture is proposed to convert images containing mathematical formulas into LaTeX code. Additionally, a new large dataset is introduced, enhancing the accuracy of img2latex performance. Compared with other models, this approach successfully addresses the major challenge of recognizing both printed and handwritten mathematical formulas within the same model. The results demonstrate state-of-the-art performance on the CROHME 2016/2019 and Im2latex-100k test sets.

The major challenge of recognizing mathematical formulas, including both printed and handwritten formulas within the same model, has been successfully addressed. Previous studies faced difficulties handling complex formulas and achieving state-of-the-art results for either handwritten or printed mathematical expressions individually. The proposed Sumen architecture, with improvements in the Vision module and the incorporation of a large dataset, has surpassed previous methods, significantly enhancing the model’s generalization ability.

This achievement clearly demonstrates the exceptional performance of the Sumen model compared to earlier approaches. The BLEU and ED metrics on the img2latex100k benchmark highlight the model’s accuracy and versatility in converting images into LaTeX format. Similarly, the WER and Exprate metrics on the CROHME benchmark set new records, showcasing the model’s flexibility and efficiency across diverse types of mathematical formulas.

However, the EM metric is relatively lower despite the highest scores for BLEU and ED. This discrepancy arises because EM evaluates individual characters, while BLEU and ED assess the overall string. Even a small number of character errors in the code can significantly impact the EM score, while the overall structure of the string remains accurate, as reflected in BLEU and ED metrics.

Future work will focus on developing a large dataset of handwritten mathematical expressions to balance the current dataset. This will involve using GANs, as proposed in [32] to convert printed mathematical expression images into handwritten mathematical expression images.

## References

- [1] <https://www.v7labs.com/open-datasets/aida>.
- [2] . <https://www.kaggle.com/datasets/rvente/im2latex170k>.
- [3] . <https://www.kaggle.com/datasets/gregoryeritsyan/im2latex-230k>.
- [4] . <https://huggingface.co/datasets/AlFrauch/im2latex>.
- [5] <https://huggingface.co/datasets/Azu/Handwritten-Mathematical-Expression-Convert-Latex>.
- [6] Katex. <https://katex.org/>.
- [7] <https://huggingface.co/datasets/Oleehy0/latex-formulas>.
- [8] 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. *CoRR*, abs/2112.03603, 2021. URL <https://arxiv.org/abs/2112.03603>.
- [9] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural optical understanding for academic documents. *CoRR*, abs/2308.13418, 2023. doi: 10.48550/ARXIV.2308.13418. URL <https://doi.org/10.48550/arXiv.2308.13418>.
- [10] Chungkwong Chan. Stroke extraction for offline handwritten mathematical expression recognition. *IEEE Access*, 8:61565–61575, 2020. doi: 10.1109/ACCESS.2020.2984627.
- [11] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. Symbolic discovery of optimization algorithms. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL [http://papers.nips.cc/paper\\_files/paper/2023/hash/9a39b4925e35cf447ccba8757137d84f-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2023/hash/9a39b4925e35cf447ccba8757137d84f-Abstract-Conference.html).

- [12] Yuntian Deng, Anssi Kanervisto, and Alexander M. Rush. What you get is what you see: A visual markup decompiler. *CoRR*, abs/1609.04938, 2016. URL <http://arxiv.org/abs/1609.04938>.
- [13] Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M. Rush. Image-to-markup generation with coarse-to-fine attention. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 980–989. PMLR, 2017. URL <http://proceedings.mlr.press/v70/deng17a.html>.
- [14] Haisong Ding, Kai Chen, and Qiang Huo. An encoder-decoder approach to handwritten mathematical expression recognition with multi-head attention and stacked decoder. In Josep Lladós, Daniel Lopresti, and Seiichi Uchida, editors, *Document Analysis and Recognition – ICDAR 2021*, pages 602–616, Cham, 2021. Springer International Publishing. ISBN 978-3-030-86331-9.
- [15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
- [16] Trung Hoang and Bao Thai. Fushion image to latex dataset, 2024. <https://www.kaggle.com/datasets/hongtrung/image-to-latex-dataset>.
- [17] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. URL <http://arxiv.org/abs/1608.06993>.
- [18] Anh Duc Le. Recognizing handwritten mathematical expressions via paired dual loss attention network and printed mathematical expressions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2413–2418, 2020. doi: 10.1109/CVPRW50498.2020.00291.

- [19] Anh Duc Le, Bipin Indurkha, and Masaki Nakagawa. Pattern generation strategies for improving recognition of handwritten mathematical expressions. *CoRR*, abs/1901.06763, 2019. URL <http://arxiv.org/abs/1901.06763>.
- [20] Anh Duc Le, Bipin Indurkha, and Masaki Nakagawa. Pattern generation strategies for improving recognition of handwritten mathematical expressions. *Pattern Recognit. Lett.*, 128:255–262, 2019. doi: 10.1016/J.PATREC.2019.09.002. URL <https://doi.org/10.1016/j.patrec.2019.09.002>.
- [21] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics. Doklady*, 10:707–710, 1965. URL <https://api.semanticscholar.org/CorpusID:60827152>.
- [22] Zhe Li, Lianwen Jin, Songxuan Lai, and Yecheng Zhu. Improving attention-based handwritten mathematical expression recognition with scale augmentation and drop attention. In *17th International Conference on Frontiers in Handwriting Recognition, ICFHR 2020, Dortmund, Germany, September 8-10, 2020*, pages 175–180. IEEE, 2020. doi: 10.1109/ICFHR2020.2020.00041. URL <https://doi.org/10.1109/ICFHR2020.2020.00041>.
- [23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, abs/2103.14030, 2021. URL <https://arxiv.org/abs/2103.14030>.
- [24] Mahshad Mahdavi, Richard Zanibbi, Harold Mouchère, 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 2019, Sydney, Australia, September 20-25, 2019*, pages 1533–1538. IEEE, 2019. doi: 10.1109/ICDAR.2019.00247. URL <https://doi.org/10.1109/ICDAR.2019.00247>.
- [25] Harold Mouchère, Christian Viard-Gaudin, Richard Zanibbi, and Utpal Garain. ICFHR 2014 competition on recognition of on-line handwritten mathematical expressions (CROHME 2014). In *14th International Conference on Frontiers in Handwriting Recognition, ICFHR*

- 2014, Crete, Greece, September 1-4, 2014, pages 791–796. IEEE Computer Society, 2014. doi: 10.1109/ICFHR.2014.138. URL <https://doi.org/10.1109/ICFHR.2014.138>.
- [26] Harold Mouchère, Christian Viard-Gaudin, Richard Zanibbi, and Utpal Garain. ICFHR2016 CROHME: competition on recognition of online handwritten mathematical expressions. In *15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016*, pages 607–612. IEEE Computer Society, 2016. doi: 10.1109/ICFHR.2016.0116. URL <https://doi.org/10.1109/ICFHR.2016.0116>.
- [27] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL, 2002. doi: 10.3115/1073083.1073135. URL <https://aclanthology.org/P02-1040/>.
- [28] Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org, 2013. URL <http://proceedings.mlr.press/v28/pascanu13.html>.
- [29] Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. Mathbert: A pre-trained model for mathematical formula understanding. *CoRR*, abs/2105.00377, 2021. URL <https://arxiv.org/abs/2105.00377>.
- [30] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015. URL <http://arxiv.org/abs/1506.02640>.
- [31] Sumeet S. Singh. Teaching machines to code: Neural markup generation with visual attention. *CoRR*, abs/1802.05415, 2018. URL <http://arxiv.org/abs/1802.05415>.
- [32] Matthias Springstein, Eric Müller-Budack, and Ralph Ewerth. Unsupervised training data generation of handwritten formulas using generative



- adversarial networks with self-attention. *CoRR*, abs/2106.09432, 2021. URL <https://arxiv.org/abs/2106.09432>.
- [33] Masakazu Suzuki, Fumikazu Tamari, Ryoji Fukuda, Seiichi Uchida, and Toshihiro Kanahori. INFTY: an integrated OCR system for mathematical documents. In *Proceedings of the 2003 ACM Symposium on Document Engineering, Grenoble, France, November 20-22, 2003*, pages 95–104. ACM, 2003. doi: 10.1145/958220.958239. URL <https://doi.org/10.1145/958220.958239>.
  - [34] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. URL <http://proceedings.mlr.press/v97/tan19a.html>.
  - [35] Thanh-Nghia Truong, Cuong Tuan Nguyen, Khanh Minh Phan, and Masaki Nakagawa. Improvement of end-to-end offline handwritten mathematical expression recognition by weakly supervised learning. In *17th International Conference on Frontiers in Handwriting Recognition, ICFHR 2020, Dortmund, Germany, September 8-10, 2020*, pages 181–186. IEEE, 2020. doi: 10.1109/ICFHR2020.2020.00042. URL <https://doi.org/10.1109/ICFHR2020.2020.00042>.
  - [36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL <http://arxiv.org/abs/1706.03762>.
  - [37] Jian Wang, Yunchuan Sun, and Shenling Wang. Image to latex with densenet encoder and joint attention. In Rongfang Bie, Yunchuan Sun, and Jiguo Yu, editors, *2018 International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI 2018, Beijing, China, October 19-21, 2018*, volume 147 of *Procedia Computer Science*, pages 374–380. Elsevier, 2018. doi: 10.1016/J.PROCS.2019.01.246. URL <https://doi.org/10.1016/j.procs.2019.01.246>.

- [38] Zelun Wang and Jyh-Charn Liu. Translating math formula images to latex sequences using deep neural networks with sequence-level training. *Int. J. Document Anal. Recognit.*, 24(1):63–75, 2021. doi: 10.1007/S10032-020-00360-2. URL <https://doi.org/10.1007/s10032-020-00360-2>.
- [39] Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. Image-to-markup generation via paired adversarial learning. In Michele Berlingerio, Francesco Bonchi, Thomas Gärtner, Neil Hurley, and Georgiana Ifrim, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part I*, volume 11051 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2018. doi: 10.1007/978-3-030-10925-7\_2. URL [https://doi.org/10.1007/978-3-030-10925-7\\_2](https://doi.org/10.1007/978-3-030-10925-7_2).
- [40] Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. Handwritten mathematical expression recognition via paired adversarial learning. *Int. J. Comput. Vis.*, 128(10):2386–2401, 2020. doi: 10.1007/S11263-020-01291-5. URL <https://doi.org/10.1007/s11263-020-01291-5>.
- [41] Jianshu Zhang, Jun Du, and Li-Rong Dai. A gru-based encoder-decoder approach with attention for online handwritten mathematical expression recognition. *CoRR*, abs/1712.03991, 2017. URL <http://arxiv.org/abs/1712.03991>.
- [42] Jianshu Zhang, Jun Du, Shiliang Zhang, Dan Liu, Yulong Hu, Jin-Shui Hu, Si Wei, and Li-Rong Dai. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognit.*, 71:196–206, 2017. doi: 10.1016/J.PATCOG.2017.06.017. URL <https://doi.org/10.1016/j.patcog.2017.06.017>.
- [43] Jianshu Zhang, Jun Du, and Lirong Dai. Multi-scale attention with dense encoder for handwritten mathematical expression recognition. *CoRR*, abs/1801.03530, 2018. URL <http://arxiv.org/abs/1801.03530>.
- [44] 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, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11076–11085. PMLR, 2020. URL <http://proceedings.mlr.press/v119/zhang20g.html>.
- [45] 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, editors, *Computer Vision – ECCV 2022*, pages 392–408, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19815-1.
  - [46] Wenqi Zhao, Liangcai Gao, Zuoyu Yan, Shuai Peng, Lin Du, and Ziyin Zhang. Handwritten mathematical expression recognition with bidirectionally trained transformer. *CoRR*, abs/2105.02412, 2021. URL <https://arxiv.org/abs/2105.02412>.
  - [47] Mingle Zhou, Ming Cai, Gang Li, and Min Li. An end-to-end formula recognition method integrated attention mechanism. *Mathematics*, 11(1), 2023. ISSN 2227-7390. doi: 10.3390/math11010177. URL <https://www.mdpi.com/2227-7390/11/1/177>.