GAP: A Grammar and Position-Aware Framework for Efficient Recognition of Multi-Line Mathematical Formulas

Zhe Yang
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
yz01@mail.ustc.edu.cn

Qi Liu*
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
qiliuql@ustc.edu.cn

Kai Zhang
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
kkzhang08@ustc.edu.cn

Shwei Tong
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
tongsw@mail.ustc.edu.cn

Enhong Chen
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
cheneh@ustc.edu.cn

ABSTRACT

Formula recognition endeavors to automatically identify mathematical formulas from images. Currently, the Encoder-Decoder model has significantly advanced the translation from image to corresponding formula markups. Nonetheless, previous research primarily concentrated on single-line formula recognition, ignoring the recognition of multi-line formulas, which presents additional challenges such as more stringent grammatical restrictions and two-dimensional positions. In this work, we present GAP (**G**rammar **A**nd **P**osition-Aware formula recognition), a comprehensive framework designed to tackle the challenges in multi-line mathematical formula recognition. First, to overcome the limitations imposed by grammar, we design a novel Grammar Aware Contrastive Learning (GACL) module, integrating complex grammar rules into the transcription model through a contrastive learning mechanism. Furthermore, primitive contrastive learning lacks clear directions for comprehending grammar rules and can lead to unstable convergence or prolonged training cycles. To enhance training efficiency, we propose Rank-Based Sampling (RBS) specialized for multi-line formulas, which guides the learning process by the importance ranking of different grammar errors. Finally, spatial location information is critical considering the two-dimensional nature of multi-line formulas. To aid the model in keeping track of that global information, we introduced a Visual Coverage (VC) mechanism that incorporates historical attention information into the image features via a parameter-free way. To validate the effectiveness of our GAP framework, we construct a new dataset Multi-Line containing 12,002 multi-line formulas and conduct extensive experiments to show the efficacy of our GAP framework in capturing grammatical rules, enhancing recognition accuracy, and enhancing training efficiency. Codes and datasets are available at https://github.com/Sinon02/GAP.

CCS CONCEPTS

• **Computing methodologies** → **Object recognition**; • **Information systems** → *Multimedia information systems*; Data mining.

KEYWORDS

Formula Recognition, Contrastive Learning, Encoder-Decoder

ACM Reference Format:
Zhe Yang, Qi Liu, Kai Zhang, Shwei Tong, and Enhong Chen. 2024. GAP: A Grammar and Position-Aware Framework for Efficient Recognition of Multi-Line Mathematical Formulas. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM '24), March 4–8, 2024, Merida, Mexico.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3616855.3635776

1 INTRODUCTION

Mathematical formulas often appear in textbooks, test papers, and scientific journals, as expressing knowledge by formulas is quite concise. However, markup languages such as LaTeX and MathML are needed to represent mathematical formulae, which makes it difficult to reuse those formulas directly. Therefore, there is a great demand for recognizing formulas directly from images in the fields like online education [19] and image retrieval.

The research about converting images into corresponding markup sequences starts from [2]. The idea of primary works [1, 5, 18, 26] is to segment the image into multiple symbols and conduct spatial analysis to identify subscripts, superscripts, and fractions. However, such methods heavily rely on the accuracy of symbol segmentation

*Corresponding author.

WSDM '24, March 4–8, 2024, Merida, Mexico

ACM ISBN 979-8-4007-0371-3/24/03…$15.00
https://doi.org/10.1145/3616855.3635776

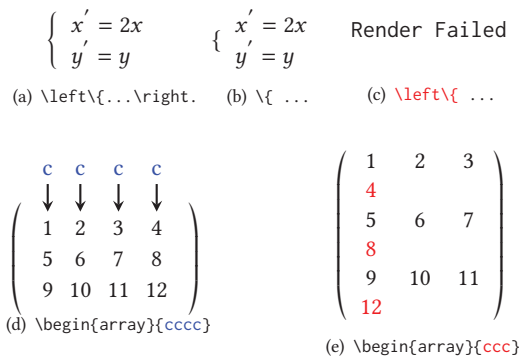

Figure 1: Examples of multi-line formula grammar rules.

and manually crafted features, limiting their recognition capability and scalability for real-world applications. With the advancement of deep neural networks, [9, 43, 45] start to adopt the sequence-to-sequence methods to replace the traditional approaches. These methods leverage the Encoder-Decoder architecture [27] to achieve end-to-end formula recognition, getting rid of symbol segmentation and manual feature engineering, resulting in significant performance improvements. However, as [32] points out, existing models perform poorly when dealing with lengthy formulas or those containing complex structures such as large matrices or nested arrays.

While prior studies have made notable progress, limited attention has been paid to the difficulties in recognizing multi-line formulas, including grammatical constraints and the distribution of two-dimensional positions. Therefore, multi-line formula recognition starts as an emerging and difficult task in the field of formula recognition. Specifically, according to our research, multi-line formula recognition presents the following new challenges compared to single-line formula recognition.

First, multi-line formulas have more stringent grammatical restrictions. For instance, multi-line formulas often require left and right brackets that can vary based on the content size. As shown in Figure 1(a), 1(b) and 1(c), certain LaTeX parsers (e.g., MathJax[1] and KaTeX[2]) require left and right operators to appear in pairs, which means redundancy on either side will cause the parsing failure. Additionally, multi-line formulas consist of rows and columns. In Figure 1(d), the {cccc} indicates that there are four columns in the array, and each column should be center-aligned. Missing even one alignment character can cause the array to change from Figure 1(d) to Figure 1(e), resulting in a completely different formula.

Second, the multi-line formulas have two-dimension positions. With rows and columns defining their structure, elements within multi-line formulas possess an added degree of freedom for placement. In fact, any equation can be an element of the multi-line formula, even the array itself. Therefore, the model not only needs to know how to translate the sub-formulas in each position but also needs to have a clear idea of its current location in two dimensions.

Finally, there are relatively few multi-line formulas in the existing public datasets. For instance, the CROHME [22] datasets (CROHME 2014, CROHME 2016, and CROHME 2019), which are the most commonly used datasets in the handwritten mathematical expression recognition (HMER) field, do not include any multi-line formulae. The IM2LATEX-100K [9] dataset, a notable resource, comprises only 6,591 formulas out of 103,556 formulas (about 6.4%) that encompass multi-line structures like matrices and tables. The scarcity of data significantly challenges the effective training of formula recognition models to comprehend multi-line formulas.

To tackle these challenges, we propose the GAP (**G**rammar **A**nd **P**osition-Aware formula recognition) framework to optimize multi-line formula recognition. The main contributions are as follows:

- To incorporate the complex grammar rules of multi-line formulas, we design the Grammar-Aware Contrastive Learning (GACL) module, which generates negative image-formula pairs by selecting translations with grammatical errors from the top-k beam search results. By maximizing the likelihood distance between positive and negative pairs, the model is trained to avoid translating grammatically incorrect sentences.
- Primitive contrastive learning lacks clear directions in understanding grammar rules, so we propose Rank-Based Sampling (RBS) specialized for multi-line formulas, which enriches the combination of positive-negative pairs by classifying different samples by the importance rankings of grammar rules.
- To overcome the two-dimensional display of multi-line equations, we propose Visual Coverage (VC) to help the model clearly remember its current spatial location and where it has already been observed by blending image features and history attention records, which is a parameter-free method.
- We construct a new dataset Multi-Line which contains 12,002 formulas while each formula contains at least one multi-line structure. Subsequently, we conduct extensive experiments on three different datasets. The results demonstrate the existing baselines can effectively capture grammatical rules, enhance recognition accuracy, and increase training efficiency after fine-tuning with the GAP framework.

2 RELATED WORKS

2.1 Formula Recognition

The end-to-end formula recognition started from [9, 45] by utilizing the attention-based encoder-decoder structure to get rid of symbol segmentation and artificial design features. Later works made improvements by updating the encoder [31, 32], changing the attention module [41, 48], and introducing GNN [35]. Although these works have significantly improved the recognition accuracy, such Encoder-Decoder models do not impose any restrictions on the generated sequence, while the markup languages are sensitive to grammatical errors. Recently, researchers have noticed the importance of grammar rules. A series of tree decoders [34, 42, 44] have been proposed to incorporate syntax information into an encoder-decoder structure. Tree decoders can capture the hierarchical tree structure of mathematical expressions, resulting in considerable improvement compared to string decoders. However, these methods are primarily designed for relatively simple syntax rules, such as superscripts and fractions, and cannot handle complex scenarios such as matrices, where multiple parts are interdependent.

Our research provides a novel perspective for incorporating grammatical information. Rather than designing a tree decoder,

[1] https://www.mathjax.org/
[2] https://katex.org/

we teach the model to avoid violating grammatical rules through contrastive learning. This allows us to add any type of syntax rule to the model, enabling us to incorporate the complex restrictions of multi-line formulas into the baseline model.

2.2 Contrastive Learning

Recently, contrastive learning (CL), as an important approach of self-supervised representation learning (SSL), has achieved significant success in the fields of computer vision [7, 11, 12, 25], natural language processing [10, 37, 38, 46], and recommender systems [6, 21, 30, 40]. The basic idea of contrastive learning is to pull an anchor sample together with positive samples and push it away from negative samples in a uniform embedding space. In self-supervised learning, positive samples typically come from data augmentation, while negative samples often encompass irrelevant instances.

However, the concept of contrastive learning extends beyond unsupervised learning. As [15] proposed, improved results can be achieved by categorizing samples into positive and negative samples based on their classification labels from supervised learning and then substituting the conventional cross-entropy loss function with the contrastive learning loss function.

Similarly, our work seeks to categorize the formulas translated by the model into positive and negative samples by applying specific grammatical rules. While similar ideas have been mentioned in the field of natural language processing [4, 39, 47], we uniquely employ these concepts to address the challenge of incorporating multi-line formula grammars into formula recognition models. Our research also introduces a novel methodology for constructing positive and negative pairs specifically designed for multi-line formulas.

2.3 Coverage Technology

The concept of Coverage originated from phrase-based statistical machine translation [17]. In machine translation [23], the coverage vector indicates whether a source phrase is translated or not, ensuring that each source phrase is translated exactly once, and alleviating issues of repetition and omission. In the image caption domain, [14] utilizes a coverage vector to indicate whether a region in the image has been observed and directs the model to pay greater attention to regions that have not been viewed. As for formula recognition field, [45] incorporates historical visual information through a CNN network; [49] proposes specialized coverage methods for transformer-based Encoder-Decoder architectures; [42] conditionally selects historical visual information based on the structure of the syntax tree structure.

As part of our training framework, we aim to seamlessly incorporate historical visual information into models lacking coverage techniques. Our proposed solution, Visual Coverage, offers a simple but highly interpretable approach. This technique maintains the model's structure and avoids the introduction of additional parameters. Instead, it integrates visual attention records into image features through an image-blending technique.

3 PRELIMINARY

3.1 Problem Definition

In this subsection, we define the formula recognition task and the multi-line formula recognition task. In formula recognition, the model is given an image containing structural components, and the objective is to generate the corresponding sequence using a specific markup language, such as LaTeX. Formally speaking, the input is an image x with width W and height H. Each pixel x_{ij} of this image has C channels, where C is always 1 to indicate a gray-scale image. The task is to generate $\hat{y} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_L\}$ that matches the given image, where L is the length of the target sequence and each y_k represents a token of markup language. If the image contains at least one multi-line structure such as an array, matrix, or table, it is considered a multi-line formula, and its corresponding recognition is multi-line formula recognition.

3.2 Transcription Model

In this section, we provide a summary of the various modifications to the Encoder-Decoder architecture that have been proposed in prior works for Formula Recognition, and these models will be trained in our proposed GAP framework.

3.2.1 Image Encoder. In the first stage of image processing, visual features are extracted by a convolutional neural network. As noted by [36], a two-dimensional feature map $V \in \mathrm{R}^{H' \times W' \times D}$ is retained to preserve structural information. Here, D represents the dimension of the feature map, and H', W' denote its height and width, respectively. To better exploit the spatial information, many studies incorporate additional positional information, which can be divided into two categories: Row Encoder and Position Encoding.

1) Row Encoder. The Row Encoder approach [9] proposes applying a recurrent neural network (RNN) sequentially across each row of the feature map V to generate a new feature grid $\hat{V}$. There are many variants of vanilla RNN, such as LSTM [13] and GRU [8], but for convenience, we use the symbol RNN to represent them in this paper. The formula for updating the RNN hidden state is $h_t = \mathrm{RNN}(h_{t-1}, x_t)$, where h_{t-1} denotes the hidden state generated by previous timestep $t-1$ and x_t indicates the input token at timestep t. In our scenario, the feature map V is updated by a RNN, i.e., for each row $h \in \{1, \cdots, H'\}$, the features at column $w \in \{1, \cdots, W'\}$ are updated by $\hat{V}_{hw} = \mathrm{RNN}(\hat{V}_{h,w-1}, V_{hw})$. The resulting feature map $\hat{V}$ now includes horizontal sequential information. To capture the vertical location information, a trainable initial hidden state $\hat{V}_{h,0}$ can be used for each row.

2) Position Encoding. The Position Encoding approach [32] tailors the 1-D positional encoding technique proposed by Transformer model [29] to a 2-D situation as follows:

$$\begin{aligned} \mathrm{PE}(x, y, 2i) &= \sin(x/10000^{4i/D}) \\ \mathrm{PE}(x, y, 2i+1) &= \cos(x/10000^{4i/D}) \\ \mathrm{PE}(x, y, 2j+D/2) &= \sin(y/10000^{4j/D}) \\ \mathrm{PE}(x, y, 2j+1+D/2) &= \cos(y/10000^{4j/D}) \end{aligned} \tag{1}$$

Here x and y denote the horizontal and vertical positions respectively, with $i, j \in \{1, \cdots, D/4\}$ specifying the dimension. The position encoding has the same shape and dimension channels as the feature map. Therefore, it can be directly added to the original feature map V to obtain a new feature map $\hat{V}$. The decisive advantage of Position Encoding is that it does not introduce any new parameters, yet it still demonstrates comparable performance to Row Encoder. Furthermore, the Position Encoding can be applied

to any shape of input image, allowing it to handle image sizes that were not seen during the training stage.

3.2.2 Markup Decoder. The Decoder's task is to generate the target sequence $\{y_t\}$ based on the feature map $\hat{V}$ from the Encoder. To address this problem, researchers commonly employ RNNs and Transformers [29], which consider prior predictions while creating new translations and can decode sequences of arbitrary lengths.

To fully exploit the visual information from feature map $\hat{V}$, attention mechanisms [3] are necessary to guide which regions require more focus when generate each predicted token. Following are three common attention methods [20]:

$$\text{Score}(h_t, h_s) = \begin{cases} h_t^T h_s & \text{dot} \\ h_t^T W_a h_s & \text{general} \\ W_a[h_t; h_s] & \text{concat} \end{cases} \tag{2}$$

Here h_t denotes the hidden state of decoder at timestep t, h_s denotes each hidden state from the encoder and $\text{Score}(h_t, h_s)$ refers to the alignment between the two hidden states. To specify which cell the model is attending to, we can introduce a latent categorical variable $z_t \in \{1, \cdots, H'\} \times \{1, \cdots, W'\}$. Here, we rewrite h_s as $\hat{V}_{hw}$ to clarify its coordinates. This allows us to write down the probability distribution of z_t and our encoder context c_t:

$$p(z_t) = \text{softmax}\left(\text{Score}\left(h_t, \hat{V}_{hw}\right)\right), \tag{3}$$

$$c_t = \sum_{h,w} p(z_t = (h, w))\hat{V}_{hw}. \tag{4}$$

At each timestep t, the previous hidden state h_{t-1} and the translation y_{t-1} are considered. Additionally, input-feeding approach [20] is used to incorporate the alignment information o_{t-1}. Therefore, the formula for updating the hidden state can be written as:

$$h_t = \text{RNN}(h_{t-1}, [y_{t-1}; o_{t-1}]), \tag{5}$$

when predicting the next token $\hat{y}_t$, we consider both the image features context c_t and the previous translation history h_t. Therefore, we concatenate them to calculate the alignment information o_t. Then, we use a softmax layer to obtain the probability distribution of the predicted token from the vocabulary, as shown below:

$$o_t = \tanh(W_c[h_t; c_t]), \tag{6}$$

$$p(\hat{y}_t | \hat{y}_1, \cdots, \hat{y}_{t-1}, \hat{V}) = \text{softmax}(W_o o_t). \tag{7}$$

Given the ground truth $y = \{y_1, y_2, \cdots, y_N\}$, the probability $P(y|x;\theta)$ to generate it from an image x using an Encoder-Decoder model with parameter θ, can be written as follows:

$$\log P(y|x;\theta) = \sum_{t=1}^{N} \log p(y_t | y_{<t}, \hat{V}; \theta). \tag{8}$$

4 THE PROPOSED GAP FRAMEWORK

In this section, we introduce our GAP framework in detail. We begin by detailing how to use contrastive learning to integrate grammatical rules into existing baseline transcription models. Then, we introduce a Rank-Based Sampling technique which can increases the diversity of positive-negative pairs based on multi-line formula grammar rules. Finally, we illustrate the design of Visual Coverage to record the history of attended regions.

4.1 Grammar-Aware Contrastive Learning

Deep neural networks excel at summarizing patterns from extensive data, enabling existing models to achieve commendable performance in formula recognition tasks without explicit external syntax information. Consider the example of a fraction, whose pattern is quite simple: `\frac{}{}`. After translating numerous fractions instances, the model can discern the need to position the numerator before the denominator to correctly construct a fraction.

```
\left(\begin{array}{cc|cc}
    1 & 2 & 3 & 4 \\
\hline 5 & 6 & 7 & 8 \\
    9 & 10 & 11 & 12 \\
\end{array}\right)
```

$\longleftrightarrow$

$$\left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ \hline 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{array}\right)$$

Figure 2: The basic grammar rules of the array.

However, summarizing patterns becomes increasingly challenging as formulas become more complex. Figure 2 displays a matrix and its corresponding notation. The basic rules for writing a grammatically correct array can be enumerated as follows:

- Begin and End: An array must start with `\begin{array}` and end with `\end{array}`, ensuring they appear in pairs.
- Aligns: The alignment specifications should directly follow the `\begin{array}` command. (1) In the brackets, alignment letters (l,c,r) dictate column justification, with the option to include the | character for vertical dividing lines; (2) The count of alignment letters should match or exceed the column count.
- Rows: After the alignment, the elements should be written row by row. (1) Optionally, use `\hline` to introduce a horizontal separator line; (2) Separate elements in adjacent columns using the & separator; (3) Each row should have the same number of elements, and any absence will be supplemented by empty placeholders; (4) Terminate each row with `\\`.

It's obvious from these grammar rules that an array must adhere to multiple regulations, and a violation of any one rule can cause the entire array to be incorrect. However, most existing Encoder-Decoder models are trained by the Teacher Forcing procedure [33], which only exposes these models to correct sequences during the training process. Therefore, it is essential for them to be aware of the cases in which their translations become grammatically incorrect.

To incorporate these syntax rules, we propose a contrastive approach that teaches the model grammar information by expanding the likelihood gap between positive image-formula pairs (ground truth) and negative image-formula pairs (contain syntax errors). As shown in Figure 3, negative pairs are generated by selecting grammatically incorrect translations from beam search outputs, because the sequences generated by beam search are top-k preferred by the model, and the model will quickly notice the importance of the grammar rules if forced not to generate these sentences.

Formally speaking, given a training set $D_T = \{\langle x^{(s)}, y^{(s)} \rangle\}_{s=1}^{S}$, where S is the total number of image-formula pairs $\langle x^{(s)}, y^{(s)} \rangle$. We first train the model using maximum likelihood estimation (MLE):

$$\hat{\theta}_{\text{MLE}} = \underset{\theta}{\text{argmin}} \{L_{NLL}(\theta)\}, \tag{9}$$

where the negative log-likelihood (NLL) is defined as:

$$L_{NLL}(\theta) = \sum_{s=1}^{S} -\log P(y^{(s)} | x^{(s)}; \theta). \tag{10}$$

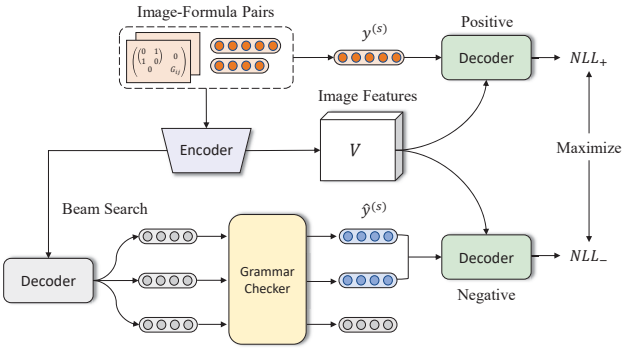

Figure 3: The Grammar-Aware Contrastive Learning module.

After the MLE training converges, the model can already generate highly accurate formulas based on images. However, these formulas may contain syntax errors. Therefore, we use the converged parameters as a starting point for Grammar-Aware Contrastive Learning (GACL) and subsequently fine-tune the model.

For each image $x^{(s)}$, we first generate top-k formula translations by beam search. Then we use a Grammar Checker to randomly select one translation $\hat{y}^{(s)}$ containing syntax errors from these predicted formulas. If such a translation $\hat{y}^{(s)}$ exists, we construct the negative image-formula pair $\langle x^{(s)}, \hat{y}^{(s)} \rangle$, and train it using the max-margin contrastive learning loss:

$$L_{CL}^{(s)} = \max\left\{\log P(\hat{y}^{(s)}|x^{(s)};\theta) + \eta - \log P(y^{(s)}|x^{(s)};\theta)), 0\right\}. \tag{11}$$

If all k predicted formulas are grammatically correct, we just set $L_{CL}^{(s)} = 0$. To ensure translation accuracy while correcting syntax errors, we combine the negative log-likelihood loss $L_{NLL}(\theta)$ with contrastive learning loss $L_{CL}(\theta)$, so the final optimization goal is:

$$\hat{\theta}_{\text{CL}} = \underset{\theta}{\text{argmin}} \left\{ \sum_{s=1}^{S} \left(L_{CL}^{(s)}(\theta) + L_{NLL}^{(s)}(\theta) \right) \right\}. \tag{12}$$

4.2 Rank-Based Sampling

Although the contrastive learning approach presented in Section 4.1 already enables the model to learn formula-related grammatical information to some extent, it is relatively inefficient to merely rely on ground truth as the only positive sample. Intuitively, if the positive sample is unique and constant, the contrastive learning loss function only blindly reduces the probability of the formula translation containing grammatical errors, and there is no clear direction for this optimization process.

To improve the training efficiency and fully utilize the available resources, we propose the **Rank-Based Sampling** method specifically designed for the multi-line formulas. Firstly, we classify formulas containing syntax errors into two types based on the seriousness of the errors, **Compile Error** and **Grammatical Error**. The former indicates that the errors will cause the whole formula to be unparseable, while the latter implies that the formula can still be parsed and used despite violating some rules. Subsequently, we prioritize the categories of error types by their importance. In particular, Compile Error destroys the whole formula, so all categories of it are assigned a Very High rank, whereas Grammatical Error can be further differentiated based on importance, as we demonstrate in Table 1. Finally, diverse pairs of positive and negative samples can be constructed based on their different error ranks. A valid positive-negative pair of samples can be created as long as the error rank of one sample is lower than the other's.

Table 1: The ranking list of syntax errors.

Error Type	Category	Importance Rank ↓
Compile Error	Mismatched Symbol	Very High
	Incorrect Structure	Very High
Grammatical Error	Missing Arrays	High
	Wrong Alignment	Medium
	Missing Lines	Low

Take Figure 4 as an example. The ground truth of Figure 4 is a two-row and one-column matrix with left curly braces. Our formula recognition model produced two translations, one committing Wrong Alignment error (Rank Low), i.e., writing two columns in the alignment section, and the other suffering from Mismatched Symbol error (Rank Very High), i.e., having only the begin command without the corresponding end command. According to the description in Section 4.1, both of them would be defined as negative samples, with the ground truth being the only positive sample. However, if we classify the positive and negative samples based on the importance of the error, then the model will be guided to prioritize the avoidance of more serious errors, and thus incrementally learn the complex grammar rules of the multi-line formulas.

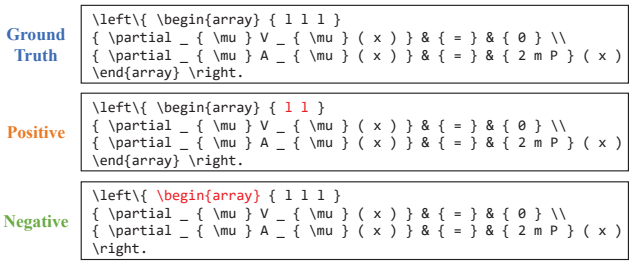

Figure 4: Example of Rank-Based Sampling.

Formally Speaking, as described in Section 4.1, we first generate k translations $\{\hat{y}_i^{(s)} | 1 \le i \le k\}$ for one sample $\langle x^{(s)}, y^{(s)} \rangle$ via beam search. Then, we classify these translations to obtain different rankings $\{r_i^{(s)} | 1 \le i \le k\}$, where $r_i^{(s)} \in$ {No Error, Low, Medium, High, Very High} as a important ascending order. Finally, we can construct a collection $D_F^{(s)}$ of positive-negative pairs :

$$D_F^{(s)} = \left\{ \langle \hat{y}_i^{(s)}, \hat{y}_j^{(s)} \rangle | 1 \le i, j \le k, i \ne j, r_i < r_j \right\}, \tag{13}$$

where $\langle x^{(s)}, \hat{y}_i^{(s)} \rangle$ is the positive sample and $\langle x^{(s)}, \hat{y}_j^{(s)} \rangle$ is the negative sample. Thus, Equation (11) can be rewritten as:

$$L_{CL}^{(s)} = \max\left\{\log P(\hat{y}_j^{(s)}|x^{(s)};\theta) + \eta - \log P(\hat{y}_i^{(s)}|x^{(s)};\theta)), 0\right\}. \tag{14}$$

4.3 Visual Coverage (VC) Mechanism

Experiments have shown that formula recognition models also suffer from over-translation and under-translation issues [14], especially in multi-line formulas. Upon analysis, we discover that certain previous models [9, 31, 32, 48, 50] only contain coverage information at the text level. Consequently, when generating translations of duplicate or similar text fragments, the model tends to lose its position in the image, resulting in repetitive translations.

As a universal training framework, we aim to incorporate image-level coverage information without introducing parameters or altering the model structure. Thus, we propose a new variable $VC_t \in \mathrm{R}^{H' \times W'}$, it accumulates the probability distribution $p(z_t) \in \mathrm{R}^{H' \times W'}$ of image features $\hat{V}$ for timesteps before t, enabling it to capture the decoding history at the image level, so we call it Visual Coverage:

$$VC_t^{(h,w)} = VC_{t-1}^{(h,w)} + p(z_{t-1} = (h, w)). \tag{15}$$

To avoid additional parameters, we combine the Visual Coverage VC_t and the image features $\hat{V}$ in an image blending way. Once integrated, the model will perceive different visual features $\hat{V}_t$ at each time step t during the decoding progress:

$$\hat{V}_t^{(h,w)} = (1 - \alpha)\hat{V}^{(h,w)} + \alpha VC_t^{(h,w)}. \tag{16}$$

Here, we use the hyper-parameter α to control the tendency between the image features and the visual coverage. Note that the introduction of VC does not alter any model equation. We only need to replace the $\hat{V}_{hw}$ with $\hat{V}_t^{(h,w)}$ in (3) and (4):

$$p(z_t) = \mathrm{softmax}\left(\mathrm{Score}\left(h_t, \hat{V}_t^{(h,w)}\right)\right), \tag{17}$$

$$c_t = \sum_{h,w} p(z_t = (h, w))\hat{V}_t^{(h,w)}. \tag{18}$$

Figure 5 presents an example of a multi-line formula that includes a nested array. With the help of Visual Coverage, we can easily know the current position is the end of the first line in the smaller array and the model will proceed to the second line of this array as it has already completed the first line translation.

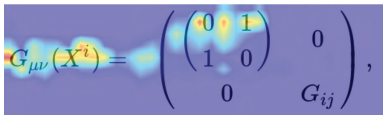

Figure 5: Visual Coverage for the multi-line formula.

5 EXPERIMENTAL SETUP

5.1 Datasets

we conduct experiments mainly on three datasets, i.e., IM2LATEX-100K, Questions and Multi-Line.

- **IM2LATEX-100K** [9] is a public dataset, it provides 103,556 different LaTeX math equations along with rendered pictures. Its train set has 83, 883 equations, validation set has 9,319 equations and test set has 10,354 equations;
- **Questions** is a private dataset, it contains 32,258 formulas extracted from real-world math questions, where the train set contains 20,624 formulas, the validation set contains 5,171 formulas and the test set contains 6,463 formulas;
- **Multi-Line** is a specific dataset for multi-line formula recognition with 12,002 formulas. The train set contains 7,190 formulas, the validation set contains 2,403 formulas and the test set contains 2,409 formulas.

Both the Questions and Multi-Line datasets are divided into 6/2/2 by length buckets with the step of 10 tokens. And all images are rendered by pdflatex[3]. The basic statistics are listed in Table 2.

Table 2: The basic statistics of the datasets.

Dataset	Image count	Multi-Line count	Avg. tokens per image	Avg. image pixels
IM2LATEX-100K	103,556	6,591	65.8	16,659.9
Questions	32,258	1,811	24.7	13,418.6
Multi-Line	12,002	12,002	44.3	12,249.2

5.2 Preprocessing

To support parallel training, it is necessary to pad similar-sized images into the same size since each formula has a different image size. As previous works [9, 32, 48] use different preprocessing strategies, we employ a unified preprocessing strategy for all baseline models, which involved cropping the image to only formula regions and then padding it to the closest $(32 \times m, 32 \times n), m, n \in \mathbb{N}^+$. After padding, we group images of the same size together. For formula texts, we divide each formula into LaTeX symbols using KaTeX, which preserves the LaTeX commands such as `\frac` as a token.

5.3 Comparison methods

To demonstrate the effectiveness of our framework, we reproduced three existing baselines as follows:

- **WYGIWYS** [9] is the first model to introduce the Encoder-Decoder structure to solve the formula recognition problem. It laid the foundation of the model for later works.
- **Double Attention** [48] make improvement to WYGIWYS by combining concat attention and dot attention.
- **MI2LS** [32] proposes the 2d position encoding mechanism to replace the Row Encoder after CNN networks.

Besides, we also compare the reported result of the following works:

- **Infty** [28] is an OCR-based mathematical expression recognition system. Its implementation InftyReader combines symbol recognition and structural analysis phases.
- **DenseNet** [31] replace the original CNN network with DensetNet and presented a novel multi-scale attention model.

5.4 Evaluation Measures

We evaluate the different models by the following metrics:

- **BLEU** [24]. BLEU evaluates the similarity of the predicted formula sequences and the ground-truth formula sequences.
- **Exact Match**. Render the predicted and ground-truth formulas back to images, and check the accuracy to exactly match.
- **Exact Match (-ws)**. The metric compares predicted images to the original images after removing whitespace columns.

[3] LaTeX (version 3.141592653-2.6-1.40.22)

Table 3: Main experimental results on the IM2LATEX-100K, Questions and Multi-Line datasets.

Dataset	Model	BLEU	Image Edit Distance	Exact Match	Exact Match (-ws)
IM2LATEX-100K	Infty	51.20	66.65	15.60	26.66
	DenseNet	88.25	-	-	-
	WYGIWYS	90.45	90.87	77.12	79.69
	+ GAP	**90.98**	**91.33**	**78.82**	**81.40**
	Double Attention	90.41	90.93	77.27	80.01
	+ GAP	90.39	91.08	77.82	80.31
	MI2LS	90.15	90.78	77.73	80.68
	+ GAP	90.44	91.28	77.96	81.14
Questions	WYGIWYS	88.52	90.91	85.77	87.03
	+ GAP	**90.41**	91.57	87.53	88.43
	Double Attention	88.89	91.10	86.61	87.58
	+ GAP	89.07	**91.67**	86.96	87.85
	MI2LS	88.74	90.27	86.91	87.71
	+ GAP	89.54	91.58	**87.76**	**88.67**
Multi-Line	WYGIWYS	90.95	87.49	82.40	83.31
	+ GAP	91.84	88.54	82.56	83.19
	Double Attention	88.06	86.18	79.74	80.90
	+ GAP	90.59	86.94	80.03	80.78
	MI2LS	90.76	86.51	82.10	83.39
	+ GAP	**91.94**	**88.64**	**84.72**	**85.72**

- **Image Edit Distance (IED)**. Binarize the predicted and the original images, then expand them to be binary strings (each pixel is represented by a number 0/1) and finally calculate the edit distance between these two strings.

5.5 Implementation Details

We reproduce three classic formula recognition models WYGIWYS [9], Double Attention [31], MI2LS [32]. The models' parameters are basically set as follows: the dimension of the image features and the decoder hidden state are both set to $D = 512$. For models containing Row Encoder, the encoder is bi-directional and its hidden state size is set to 256. The size of the token embedding is uniformly set to 80. The batch sizes for the MLE and CL stages are set to 20. Empirically, we set α to 0.3 for the Visual Coverage module.

During the MLE training phase, we start the Adam optimizer [16] from the initial learning rate 1e-3. We train all models within 30 epochs for the IM2LATEX-100K dataset, and 40 epochs for the Questions and Multi-Line datasets. For CL stage, we still use the Adam optimizer, which starts from 5e-5. We use MLE pre-trained models with the highest BLEU scores for further training, and all models are trained for 30 epochs on IM2LATEX-100K, Questions and Multi-Line datasets. All models are implemented by PyTorch and trained on a Linux server with four RTX 3090 GPUs.

6 EMPIRICAL RESULTS

In this section, we conduct extensive experiments to demonstrate the effectiveness of our framework from various aspects:

- **RQ1:** What is GAP's overall performance against baselines?
- **RQ2:** How does each component (GACL & VC) of GAP really contribute to the model performance?
- **RQ3:** How does RBS affect the contrastive learning process?
- **RQ4:** Are there some typical cases that demonstrate why the GAP framework works?

6.1 Overall Performance (RQ1)

From the results in Table 3. we can get several observations. Firstly, despite the multi-line formulas constituting only 6.4% and 5.6% of the IM2LATEX-100K and Questions comprehensive datasets respectively, employing our framework still achieves marginal enhancements. Especially for the WYGIWYS, which gains 0.53 BLEU, 0.58 IED, 1.69 EM and 1.71 EM(-ws). This proves, 1) the historical visual information also benefits the recognition of single-line formulas; 2) The grammar rules introduced by GACL can also improve the quality of single-line formula generation.

Secondly, on the Multi-Line dataset, our framework achieved great performance gains (almost 2 percentage points of metrics in general for MI2LS), especially for the image-level metrics. As previously emphasized, multi-line formulas exhibit high sensitivity to syntax, thus even minor modifications to grammar units can significantly impact the overall layout. Consequently, the increase in IED and EM indicates that the model is more likely to generate grammatically correct formulas after training in our framework. Finally, our framework demonstrates performance improvements across all three baselines, indicating its general effectiveness.

6.2 Ablation Study (RQ2)

For the sake of page limitations, in this section we will present ablation experiments results for different modules from two perspectives: fix a model and vary the datasets, and fix a dataset and vary the models. Let us start with the ablation results of the model MI2LS on the three datasets as shown in Table 4.

Table 4 demonstrates the efficacy of both the VC and GACL modules in enhancing multiple model metrics. GACL notably enhances image-level metrics like IED and EM (-ws) more than the text-level metric BLEU. However, the GACL module shows comparatively less improvement in metrics compared to the VC module.

To further investigate the impact of the GACL module, we conducted additional ablation experiments on the Multi-Line dataset

Table 4: Ablation study for MI2LS on three datasets.

Dataset	VC	GACL	BLEU	IED	EM	EM (-ws)
IM2LATEX-100K	✗	✗	90.15	90.78	77.73	80.68
	✓	✗	90.38	91.01	77.70	80.90
	✓	✓	**90.44**	**91.28**	**77.96**	**81.14**
Questions	✗	✗	88.74	90.27	86.91	87.71
	✓	✗	89.21	91.01	**87.88**	88.63
	✓	✓	**89.54**	**91.58**	87.76	**88.67**
Multi-Line	✗	✗	90.76	86.51	82.10	83.39
	✓	✗	91.91	88.11	**84.97**	84.97
	✓	✓	**91.94**	**88.64**	84.72	**85.72**

to assess the performance variations of the three baseline models. We introduced new conditions that only utilized the GACL module, and the experimental results are presented in Table 5.

Based on Table 5, the following conclusions can be drawn. Firstly, the GACL module consistently improves image-level metrics, regardless of its integration with the VC module. This may be due to GACL's training objective, which aims not only to align model output formula with ground truth but also to ensure adherence to grammatical rules. This goal inherently conflicts with text-level metrics. Additionally, the impact of GACL on metrics depends on the base model. For example, GACL produces more significant improvements for MI2LS than Double Attention and shows greater enhancements for models combined with VC. This suggests that stronger baselines allow GACL to provide models with a deeper understanding of grammar information. In general, models finetuned with GACL produce higher-quality recognition results.

6.3 Effect of Rank-Based Ranking (RQ3)

In this section, we explore the impact of Rank-Based Sampling (RBS). First, the MI2LS model is trained for 40 epochs using MLE method on the Multi-Line dataset. Then, the model with the highest BLEU score is selected as the starting point. Finally, the model is trained using the vanilla contrastive learning and the contrastive learning with the RBS strategy with different hyperparameter η.

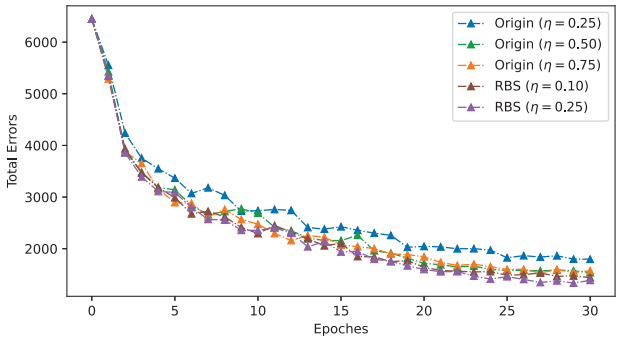

Figure 6: The Effect of Rule-Based Ranking.

In Figure 6, we track the occurrence of grammatical errors among the top-k potential translation results throughout the training process. Based on the experimental results, it is evident that the training curve based on Rule-Based Sampling demonstrates a more rapid reduction in errors, ultimately converging to fewer potential errors. Additionally, the hyperparameter η has a certain impact on the training process. A small η can weaken contrastive learning, resulting in an increased occurrence of errors.

Table 5: Ablation study for Multi-Line on three models.

Model	VC	GACL	BLEU	IED	EM	EM (-ws)
WYGIWYS	✗	✗	90.95	87.49	82.40	83.31
	✓	✗	**92.28**	87.83	82.06	82.89
	✗	✓	90.81	87.71	82.48	**83.44**
	✓	✓	91.84	**88.54**	**82.56**	83.19
Double Attention	✗	✗	88.06	86.18	79.74	**80.90**
	✓	✗	**90.87**	86.48	79.83	80.78
	✗	✓	88.29	86.38	79.36	80.53
	✓	✓	90.59	**86.94**	**80.03**	80.78
MI2LS	✗	✗	90.76	86.51	82.10	83.39
	✓	✗	91.91	88.11	84.01	84.97
	✗	✓	90.74	86.83	82.27	83.60
	✓	✓	**91.94**	**88.64**	**84.72**	**85.72**

6.4 Case Study (RQ4)

In this section, we analyze a practical case to illustrate why our GAP framework can optimize the recognition results.

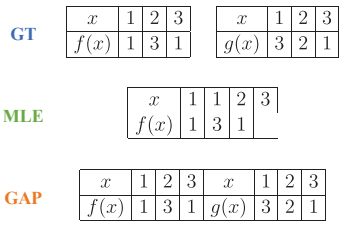

Figure 7: Comparison of recognition results.

As shown in Figure 7, the MLE-trained model made two mistakes: 1) it didn't identify the existence of two arrays in the original image; 2) it produced repeated translations when translating the first line, resulting in one less element in the second line. After being finetuned by the GAP framework, the model now has a global visual memory, so although the model still only generates one array, it fully restores the information in the original image. Moreover, the incorporation of syntax rules prevents the occurrence of vacancies resulting from varying element counts across different rows.

7 CONCLUSION

In this paper, we presented a comprehensive framework GAP to address the challenges in multi-line formula recognition. GAP skillfully incorporates syntax rules, which are hard to be handled by tree-decoders, into the baseline models via a contrastive learning approach, and achieves promising improvements in different metrics. Additionally, in order to enhance the diversity of positive and negative sample pairs, we also proposed Rule-Based Samplig to guide the model to learn grammar rules according to their importance. Finally, GAP also introduces visual coverage through a simple parameter-free mechanism, and gains decent effects. We hope this work will lead to more studies in the future.

ACKNOWLEDGMENTS

This research was partial supported by grants from the National Key Research and Development Program of China (Grant No. 2021YFF0901003), the Anhui Provincial Natural Science Foundation (No. 2308085QF229), and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Francisco Álvaro, Joan-Andreu Sánchez, and José-Miguel Benedí. 2014. Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models. *Pattern Recognition Letters* 35 (2014), 58–67.

[2] Robert H Anderson. 1967. Syntax-directed recognition of hand-printed two-dimensional mathematics. In *Symposium on interactive systems for experimental applied mathematics: Proceedings of the Association for Computing Machinery Inc. Symposium*. 436–459.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).

[4] Hannan Cao, Wenmian Yang, and Hwee Tou Ng. 2021. Grammatical Error Correction with Contrastive Learning in Low Error Density Domains. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. 4867–4874.

[5] Kam-Fai Chan and Dit-Yan Yeung. 2001. Error detection, error correction and performance evaluation in on-line mathematical expression recognition. *Pattern Recognition* 34, 8 (2001), 1671–1684.

[6] Mengru Chen, Chao Huang, Lianghao Xia, Wei Wei, Yong Xu, and Ronghua Luo. 2023. Heterogeneous graph contrastive learning for recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 544–552.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[9] Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M Rush. 2017. Image-to-markup generation with coarse-to-fine attention. In *International Conference on Machine Learning*. PMLR, 980–989.

[10] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 6894–6910.

[11] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems* 33 (2020), 21271–21284.

[12] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9729–9738.

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

[14] Teng Jiang, Zehan Zhang, and Yupu Yang. 2019. Modeling coverage with semantic embedding for image caption generation. *The Visual Computer* 35 (2019), 1655–1665.

[15] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems* 33 (2020), 18661–18673.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. 127–133.

[18] Stephane Lavirotte and Loic Pottier. 1998. Mathematical formula recognition using graph grammar. In *Document Recognition V*, Vol. 3305. SPIE, 44–52.

[19] Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, and Guoping Hu. 2019. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering* 33, 1 (2019), 100–115.

[20] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1412–1421.

[21] Jianxin Ma, Chang Zhou, Hongxia Yang, Peng Cui, Xin Wang, and Wenwu Zhu. 2020. Disentangled self-supervision in sequential recommenders. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 483–491.

[22] Mahshad Mahdavi, Richard Zanibbi, Harold Mouchere, Christian Viard-Gaudin, and Utpal Garain. 2019. 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)*. IEEE, 1533–1538.

[23] Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. 2016. Coverage Embedding Models for Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 955–960.

[24] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[26] Shinji Sako, Takuya Nishimoto, Shigeki Sagayama, et al. 2006. On-line recognition of handwritten mathematical expressions based on stroke-based stochastic context-free grammar. In *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft.

[27] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems* 27 (2014).

[28] Masakazu Suzuki, Fumikazu Tamari, Ryoji Fukuda, Seiichi Uchida, and Toshihiro Kanahori. 2003. Infty: an integrated ocr system for mathematical documents. In *Proceedings of the 2003 ACM symposium on Document engineering*. 95–104.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[30] Hao Wang, Yao Xu, Cheng Yang, Chuan Shi, Xin Li, Ning Guo, and Zhiyuan Liu. 2023. Knowledge-Adaptive Contrastive Learning for Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 535–543.

[31] Jian Wang, Yunchuan Sun, and Shenling Wang. 2019. Image to latex with densenet encoder and joint attention. *Procedia computer science* 147 (2019), 374–380.

[32] Zelun Wang and Jyh-Charn Liu. 2021. Translating math formula images to LaTeX sequences using deep neural networks with sequence-level training. *International Journal on Document Analysis and Recognition (IJDAR)* 24, 1 (2021), 63–75.

[33] Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation* 1, 2 (1989), 270–280.

[34] Changjie Wu, Jun Du, Yunqing Li, Jianshu Zhang, Chen Yang, Bo Ren, and Yiqing Hu. 2022. TDv2: A Novel Tree-Structured Decoder for Offline Mathematical Expression Recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 2694–2702.

[35] Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. 2021. Graph-to-graph: towards accurate and interpretable online handwritten mathematical expression recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 2925–2933.

[36] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. PMLR, 2048–2057.

[37] Ziyun Xu, Chengyu Wang, Minghui Qiu, Fuli Luo, Runxin Xu, Songfang Huang, and Jun Huang. 2023. Making pre-trained language models end-to-end few-shot learners with contrastive prompt tuning. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 438–446.

[38] Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741* (2021).

[39] Zonghan Yang, Yong Cheng, Yang Liu, and Maosong Sun. 2019. Reducing Word Omission Errors in Neural Machine Translation: A Contrastive Learning Approach. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 6191–6196.

[40] Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H Chi, Steve Tjoa, Jieqi Kang, et al. 2021. Self-supervised learning for large-scale item recommendations. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4321–4330.

[41] Yu Yin, Zhenya Huang, Enhong Chen, Qi Liu, Fuzheng Zhang, Xing Xie, and Guoping Hu. 2018. Transcribing content from structural images with spotlight mechanism. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2643–2652.

[42] Ye Yuan, Xiao Liu, Wondimu Dikubab, Hui Liu, Zhilong Ji, Zhongqin Wu, and Xiang Bai. 2022. Syntax-Aware Network for Handwritten Mathematical Expression Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4553–4562.

[43] Jianshu Zhang, Jun Du, and Lirong Dai. 2018. Multi-scale attention with dense encoder for handwritten mathematical expression recognition. In *2018 24th international conference on pattern recognition (ICPR)*. IEEE, 2245–2250.

[44] Jianshu Zhang, Jun Du, Yongxin Yang, Yi-Zhe Song, Si Wei, and Lirong Dai. 2020. A tree-structured decoder for image-to-markup generation. In *International Conference on Machine Learning*. PMLR, 11076–11085.

[45] Jianshu Zhang, Jun Du, Shiliang Zhang, Dan Liu, Yulong Hu, Jinshui Hu, Si Wei, and Lirong Dai. 2017. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognition* 71 (2017), 196–206.

[46] Kai Zhang, Qi Liu, Hao Qian, Biao Xiang, Qing Cui, Jun Zhou, and Enhong Chen. 2021. Eatn: An efficient adaptive transfer network for aspect-level sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering* 35, 1 (2021), 377–389.

[47] Kai Zhang, Kun Zhang, Mengdi Zhang, Hongke Zhao, Qi Liu, Wei Wu, and Enhong Chen. 2022. Incorporating dynamic semantics into pre-trained language model for aspect-based sentiment analysis. *arXiv preprint arXiv:2203.16369* (2022).

[48] Wei Zhang, Zhiqiang Bai, and Yuesheng Zhu. 2019. An improved approach based on CNN-RNNs for mathematical expression recognition. In *Proceedings of the 2019 4th international conference on multimedia systems and signal processing*. 57–61.

[49] Wenqi Zhao and Liangcai Gao. 2022. Comer: Modeling coverage for transformer-based handwritten mathematical expression recognition. In *European Conference on Computer Vision*. Springer, 392–408.

[50] Mingle Zhou, Ming Cai, Gang Li, and Min Li. 2023. An End-to-End Formula Recognition Method Integrated Attention Mechanism. *Mathematics* 11, 1 (2023), 177.