Handwritten Mathematical Expression Recognition via Attention Aggregation Based Bi-directional Mutual Learning

Xiaohang Bian[1], Bo Qin [2], Xiaozhe Xin [2], Jianwu Li [1*], Xuefeng Su [2], Yanfeng Wang[2],

[1] School of Computer Science and Technology, Beijing Institute of Technology, China
[2] AI Interaction Department, Tencent, China
{xhang.bian,ljw}@bit.edu.cn, {bqin,georgexin,felixsu,shawndwang}@tencent.com,

Abstract

Handwritten mathematical expression recognition aims to automatically generate LaTeX sequences from given images. Currently, attention-based encoder-decoder models are widely used in this task. They typically generate target sequences in a left-to-right (L2R) manner, leaving the right-to-left (R2L) contexts unexploited. In this paper, we propose an Attention aggregation based Bi-directional Mutual learning Network (ABM) which consists of one shared encoder and two parallel inverse decoders (L2R and R2L). The two decoders are enhanced via mutual distillation, which involves one-to-one knowledge transfer at each training step, making full use of the complementary information from two inverse directions. Moreover, in order to deal with mathematical symbols in diverse scales, an Attention Aggregation Module (AAM) is proposed to effectively integrate multi-scale coverage attentions. Notably, in the inference phase, given that the model already learns knowledge from two inverse directions, we only use the L2R branch for inference, keeping the original parameter size and inference speed. Extensive experiments demonstrate that our proposed approach achieves the recognition accuracy of 56.85 % on CROHME 2014, 52.92 % on CROHME 2016, and 53.96 % on CROHME 2019 without data augmentation and model ensembling, substantially outperforming the state-of-the-art methods. The source code is available in https://github.com/XH-B/ABM.

Introduction

Handwritten Mathematical Expression Recognition (HMER) has multiple application scenarios, such as intelligent education, human-computer interaction and academic paper writing auxiliary tools. Traditional methods generating LaTeX sequences from input images always depend on specially designed grammars (Lavirotte and Pottier 1998; Chan and Yeung 2001; MacLean and Labahn 2013). These grammars need extensive prior knowledge to define mathematical expression structures, symbols' position relationship, and corresponding parsing algorithms, so that they cannot recognize complex mathematical expressions.

Recently, attention based encoder-decoder models have been applied to HMER, due to its excellence in machine translation (Cho et al. 2014), speech recognition (Bahdanau et al. 2016), segmentation (Tianfei et al. 2020, 2021; Wenguan et al. 2021). These attention-based methods (Deng et al. 2017; Le and Nakagawa 2017; Zhang et al. 2017, 2018; Zhang, Du, and Dai 2018; Wu et al. 2018; Le, Indurkhya, and Nakagawa 2019; Li et al. 2020; Wu et al. 2020) are remarkably superior to grammar-based ones. For example, WAP (Zhang et al. 2017) first introduces the 2D coverage attention to solve the problem of lacking coverage, as shown in Fig. 1 (a). The coverage attention is the sum of past attentions aiming to keep track of past alignment information, such that the attention model can be guided to assign higher attention probabilities to the untranslated regions of images. Nevertheless, one main limitation of the coverage attention is that it only employs historical alignment information, disregarding future information (untranslated regions). For example, many mathematical expressions are of symmetrical structures, where the left "{" and right "}" braces always appear together or sometimes far apart. And some symbols in an equation are correlated, such as "$\int$" and "dx". Most methods only use the left-to-right coverage attention to identify the current symbol, ignoring the fact that the future information from the right is also important, which may cause the problem of attention drift. And the captured dependence information between the current symbol and previous symbols becomes weaker as their distances increase. Therefore, they may not adequately exploit long-distance correlation or grammatical specification of mathematical expressions (Zhao et al. 2021). BTTR (Zhao et al. 2021) uses a transformer decoder with two directions to handle attention drift (Fig. 1(f)), but there is no explicitly supervised information for BTTR to learn from the reversed direction, and BTTR aligns attention without a coverage mechanism, making it still suffer from some limitations in recognizing long formulas. Besides, variable scales of characters in a mathematical expression may result in recognition difficulty or uncertainty. DWAP-MSA (Zhang et al. 2018) attempts to encode multi-scale features to alleviate this problem. However, they do not scale the local receptive field, but only scale the feature map, making it impossible to accurately attend to small characters during recognition.

Thereby, we propose a novel framework with **A**ttention aggregation and **B**i-directional **M**utual learning (ABM) for HMER, as shown in Fig. 1(g). Specially, our framework includes three modules: Feature Extraction, Attention Aggre-

*Corresponding author: *Jianwu Li*.

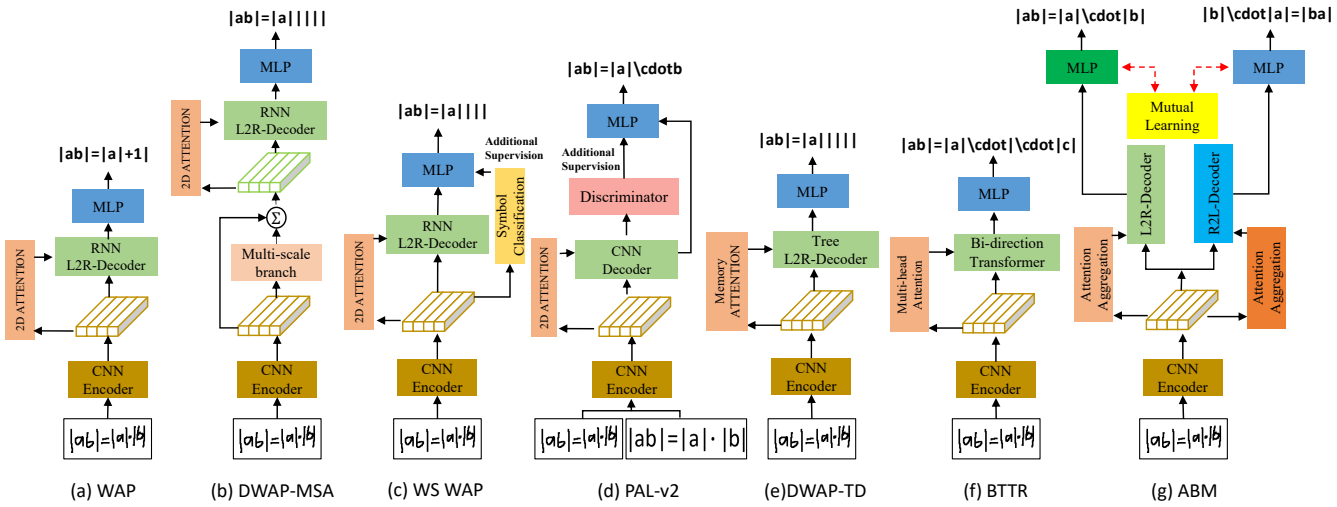

Figure 1: Typical architectures and our proposed model for HMER. (a) is 2D attention based encoder-decoder framework for HMER (Zhang et al. 2017). (b) uses multi-scale features (Zhang et al. 2018). (c) introduces a symbol classification network (Truong et al. 2020). (d) utilizes existing printed expressions (Wu et al. 2020). (e) introduces a tree decoder (Zhang et al. 2020). (f) uses a transformer decoder (Zhao et al. 2021). (g) is our proposed model with two inverse decoders learning from each other to enhance their decoding ability.

gation and Bi-directional Mutual Learning. (1) In Feature Extraction module, we use DenseNet as feature extractor as it has proved to be effective in WAP (Zhang et al. 2017). (2) In Attention Aggregation module, we propose multi-scale coverage attention to recognize characters of different sizes in mathematical expressions, thereby improving the recognition accuracy at the current moment and alleviating the problem of error accumulation. (3) In Bi-directional Mutual Learning module, we propose a novel decoder framework with two parallel decoder branches in opposite decoding directions (L2R and R2L) and use mutual distillation to learn from each other. Specifically, this framework helps the current coverage attention to capitalize upon historical and future information sufficiently at the same time, so as to better determine the current attention position. Therefore, each branch can learn more complementary context information and explore long-distance dependency information through step-by-step mutual learning, leading to stronger decoding ability. Note that while we use two decoders for training, we only use one L2R branch for inference. Our contributions are summarized as follows:

(1) We propose a novel bi-directional mutual learning framework with a shared encoder and two inverse decoders, to better learn complementary context information. To our best knowledge, we are the first to introduce mutual learning into HMER.

(2) We propose a multi-scale coverage attention mechanism to better recognize the symbols with variable scales in an expression.

(3) Comprehensive experiments show that ABM greatly surpasses the state-of-the-art methods on CROHME 2014, 2016 and 2019, respectively. And the ABM framework can be applied to various decoders including GRU, LSTM, and Transformer.

Related Work

Methods of HMER Traditional HMER methods require specially designed grammatical rules to represent the two-dimensional structural information of formulas, such as graph grammar (Lavirotte and Pottier 1998), attributive clause grammar (Chan and Yeung 2001), relational grammar (MacLean and Labahn 2013) or probabilistic model based grammar (Awal, Mouchere, and Viard-Gaudin 2014; MacLean and Labahn 2015; Álvaro, Sánchez, and Benedí 2016).

In recent years, with the success of sequence learning in various applications, such as machine translation (Luong et al. 2014) and speech recognition (Bahdanau et al. 2016), the encoder-decoder framework has been widely used to solve image-to-sequence tasks. (Deng et al. 2017) first introduced such a framework to HEMR, which uses Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) (Kawakami 2008) as an encoder for feature extraction, while the Gated Recurrent Unit (GRU) (Chung et al. 2014) is used as a decoder to recognize LaTeX characters. Many approaches typically improve the encoder with stronger convolutional networks to strengthen feature extraction, such as introducing full convolutional networks (Zhang et al. 2017; Wang et al. 2019), DenseNet (Zhang, Du, and Dai 2018; Le, Indurkhya, and Nakagawa 2019; Truong et al. 2020) and ResNet (Li et al. 2020; Yan et al. 2021). For the decoder, most methods design attention mechanisms to improve their translation. For example, the coverage attention mechanism (Zhang et al. 2017; Zhang, Du, and Dai 2018; Li et al. 2020; Le 2020; Truong

et al. 2020), overcomes the under-parsing or over-parsing problem by considering past alignment probabilities. Additionally, some methods improve their recognition performance by introducing additional data such as data augmentation, including the pattern generation strategy PGS (Le, Indurkhya, and Nakagawa 2019), random scale enhancement SCDA (Li et al. 2020), as well as the use of printed expressions DLA (Le 2020), PAL (Wu et al. 2018) and PAL-v2 (Wu et al. 2020). Besides, some methods also improve the decoder with transformer to utilize bi-directional information (BTTR) (Zhao et al. 2021) or tree decoder to enhance the decoding ability to handle complex formulas (DWAP-TD) (Zhang et al. 2020). The structures of some models above are illustrated in Fig. 1(a∼f).

Mutual Learning Mutual learning refers to the process in which a group of models learn together during training. DML (Zhang et al. 2018) first proposed the concept of mutual learning in the field of knowledge distillation, and improved the model generalization ability through collaborative training. Co-distillation (Anil et al. 2018) forces each network to maintain its diversity through distillation losses. ONE (Lan, Zhu, and Gong 2018) trains a multi-branch network and uses the predictions of these branches as soft goals to guide each branch network. CLNN (Song and Chai 2018) designs hierarchical multiple branches and uses corresponding zoom gradients. KDCL (Guo et al. 2020) proposes to integrate soft targets from multiple networks, and then supervise the learning of each network. These methods have been widely tried on public classification datasets, and each mutual learning work is usually of single CNN-based architecture.

However, our experimental results show that direct mutual learning between two encoder-decoder networks cannot achieve satisfactory recognition accuracy for the HMER task. Therefore, we design a novel architecture consisting of a shared encoder and two inverse decoders that can be aligned in each decoding step and learn from each other to fully explore the specific features of mathematical expressions, such as symmetry and long-distance correlation.

Method

We propose a novel end-to-end architecture with Attention aggregation and Bi-directional Mutual learning (ABM) for HMER, as shown in Fig. 2. It mainly consists of three modules: 1) **F**eature **E**xtraction **M**odule (FEM) that extracts feature information from a mathematical expression image. 2) **A**ttention **A**ggregation **M**odule (AAM), which integrates multi-scale coverage attentions to align historical attention information, and effectively aggregates different scales of features from various sizes of symbols in the decoding phase. 3) **B**i-directional **M**utual **L**earning Module (BML) is comprised of two parallel decoders with opposite decoding directions (L2R and R2L) to complement information reciprocally. During training, each decoder branch can learn not only the ground-truth LaTeX sequence but also the prediction of the other branch, thereby enhancing the decoding ability.

Feature Extraction Module We use the densely connected convolutional network (DenseNet (Huang et al. 2017)) as the encoder to extract features from an input image, similar to (Zhang, Du, and Dai 2018). The output is a three-dimensional feature map $\mathcal{F}$ of $H \times W \times D$, where H, W and D respectively denote height, width and channel of the feature map. Specially, we consider the output features as content information $\mathbf{a}$ of M dimensions, forming a vector $\mathbf{a} = \{a_1, a_2, ..., a_M\}$, where $a_i \in \mathbb{R}^{\mathbb{D}}, M = H \times W$.

Attention Aggregation Module The attention mechanism prompts a decoder to focus on a specific area of input image. Specially, coverage-based attention can better track alignment information and guide a model to assign higher attention probabilities to untranslated regions (Zhang et al. 2017). Inspired by the Inception module (Szegedy et al. 2015), we propose the Attention Aggregation Module (AAM) to aggregate different receptive fields on coverage attention. Compared with traditional attention, AAM not only pays attention to the detailed features of local areas, but also the global information on larger receptive fields. Therefore, AAM will generate finer information alignment and help the model capture more accurate spatial relationships. Note that our AAM is different from DWAP-MSA (Zhang et al. 2018), which proposes a dense encoder with multi-scale branches to generate low-resolution and high-resolution features, requiring more parameters and calculations. Fig. 2 shows the details of our AAM that uses the hidden state $\hat{h}_t$, feature map $\mathcal{F}$ and coverage attention β_t to compute the current attention weights α_t and then obtain context vector c_t:

$$A_s = U_s\beta_t, A_l = U_l\beta_t, \tag{1}$$

U_s and U_l denote the convolution operations of small and large kernel sizes ($e.g.$, 5, 11), respectively, and β_t represents the sum of all past attention probabilities, which is initialized as a zero vector and then calculated by

$$\beta_t = \sum_{l=1}^{t-1} \alpha_l, \tag{2}$$

where α_l denotes the attention score at step l. The current attention map α_t is calculated by

$$\alpha_t = v_a^T \tanh(W_{\hat{h}}\hat{h}_t + U_f\mathcal{F} + W_sA_s + W_lA_l), \tag{3}$$

where $W_{\hat{h}} \in \mathbb{R}^{n\times d}$, $W_s \in \mathbb{R}^{1\times d}$ and $W_l \in \mathbb{R}^{1\times d}$ are trainable weight matrices, U_f is 1×1 convolution operation, and $\hat{h}_t$ denotes the hidden state generated from a GRU in Eq. (7).

The context vector is denoted as c_t and is computed as a weighted sum of the feature content information $\mathbf{a}$:

$$c_t = \sum_{i=1}^{M} \alpha_{t,i}\mathbf{a}_i, \tag{4}$$

where $\alpha_{t,i}$ is the weight of the i-th feature of $\mathcal{F}$ at step t.

Bi-directional Mutual Learning Module Given an input mathematical expression image, traditional HMER methods decode it from left to right (L2R) (Zhang et al. 2017; Zhang, Du, and Dai 2018), without sufficiently considering the long-distance dependence. Therefore, we propose to leverage dual-stream decoders to translate the input image to LaTex sequences in two opposite directions (L2R and R2L),

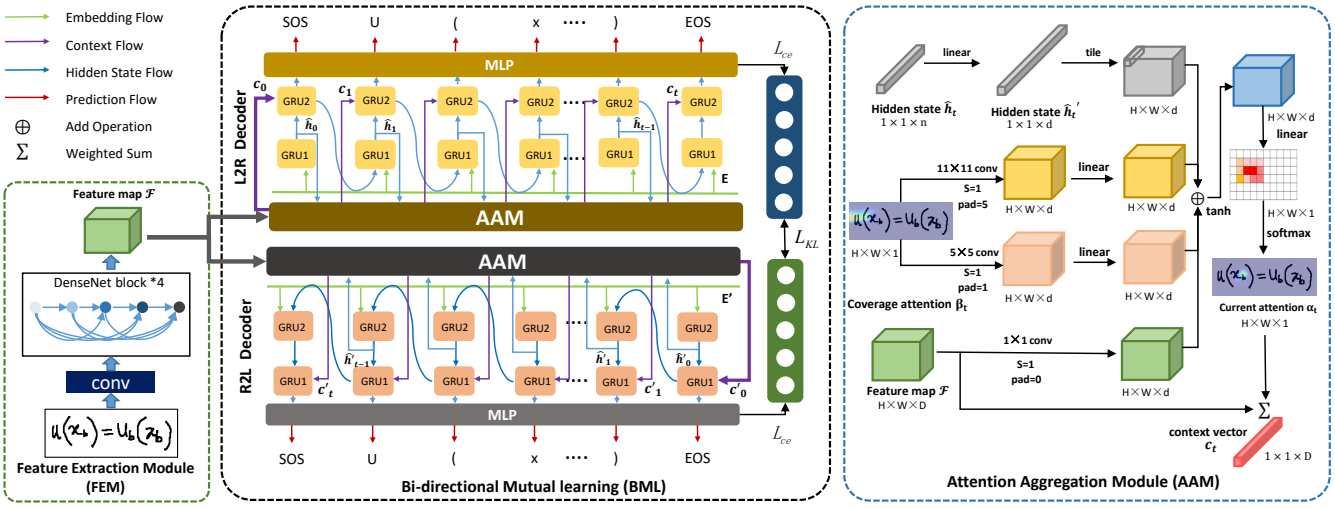

Figure 2: Architecture of our proposed model. An image is first input into the DenseNet to extract features, then two decoders separately generate the LaTeX sequences in two reverse directions. During decoding, two branches are trained by minimizing the distance between their predicted probabilities at each time step. An attention aggregation module is proposed to generate current attention by coverage attentions with different scales. The operation "tile" duplicates the hidden state vector $H \times W$ times. The "MLP" denotes the trainable multi-layer perception layers. The two branches do not share parameters during training.

and then learn the decoding information from each other. The two branches have the same architecture, but are merely different in their decoding directions.

For bi-directional training, we add $\langle sos \rangle$ and $\langle eos \rangle$ respectively as the start and end symbol of LaTeX sequences. Specially, for the target LaTeX sequence of length T, $\mathcal{Y} = \{Y_1, Y_2, ..., Y_T\}$, we denote it from left to right (L2R) as $\overrightarrow{\mathcal{Y}}_{l2r} = \{\langle sos \rangle, Y_1, Y_2, ..., Y_T, \langle eos \rangle\}$ and from right to left (R2L) as $\overleftarrow{\mathcal{Y}}_{r2l} = \{\langle eos \rangle, Y_T, Y_{T-1}, ..., Y_1, \langle sos \rangle\}$. The probabilities of the predicted symbols at step t for the L2R and R2L branch, are computed as follows:

$$p(\overrightarrow{y}_t | \overrightarrow{y}_{t-1}) = W_o max(W_y E \overrightarrow{y}_{t-1} + W_h h_t + W_t c_t), \quad (5)$$

$$p(\overleftarrow{y}_t | \overleftarrow{y}_{t-1}) = W_o^{'} max(W_y^{'} E^{'} \overleftarrow{y}_{t-1} + W_h^{'} h_t^{'} + W_t^{'} c_t^{'}), \quad (6)$$

where h_t, $\overrightarrow{y}_t$ denote the current state and previous prediction output at step t in the L2R branch. The mark $*^{'}$ denotes the R2L branch. $W_o \in \mathbb{R}^{K \times d}$, $W_y \in \mathbb{R}^{d \times n}$, $W_h \in \mathbb{R}^{d \times n}$ and $W_t \in \mathbb{R}^{d \times D}$ are trainable weight matrices. Let d, K and n denote the attention dimension, the number of symbol classes and GRU dimension, respectively. E is an embedding matrix. max denotes the maxout activation function. The hidden representations $\{h_1, h_2, ..., h_t\}$ are produced by:

$$\hat{h_t} = f_1(h_{t-1}, E \overrightarrow{y}_{t-1}), \quad (7)$$

$$h_t = f_2(\hat{h_t}, c_t), \quad (8)$$

where f_1 and f_2 denote two unidirectional GRU models similar to (Zhang et al. 2017).

We define the probability of the L2R branch as $\overrightarrow{\mathcal{P}}_{l2r} = \{\langle sos \rangle, \overrightarrow{y}_1, \overrightarrow{y}_2, ..., \overrightarrow{y}_T, \langle eos \rangle\}$, and R2L branch as $\overleftarrow{\mathcal{P}}_{r2l} = \{\langle eos \rangle, \overleftarrow{y}_1, \overleftarrow{y}_2, ..., \overleftarrow{y}_T, \langle sos \rangle\}$, where $\overrightarrow{y}_i \in \mathbb{R}^K$ is the predicted probability of label symbols when the i-th step decoding is performed. In order to apply mutual learning to the prediction distributions from two branches, we need to align the LaTeX sequences generated by the L2R and R2L decoders. Specifically, we discard the first and last predictions ($\langle eos \rangle$ and $\langle sos \rangle$) to obtain $\overrightarrow{\mathcal{P}}_{l2r}^{'}$ and $\overleftarrow{\mathcal{P}}_{r2l}^{'}$, and then reverse $\overleftarrow{\mathcal{P}}_{r2l}^{'}$ to obtain $\overleftarrow{\mathcal{P}}_{r2l}^{*} = \{\overleftarrow{y}_T, \overleftarrow{y}_{T-1}, ..., \overleftarrow{y}_1\}$. At the same time, Kullback-Leibler (KL) loss is introduced to quantify the difference in prediction distribution between them. During training, we use the soft probabilities generated by the model to provide more information, similar to (Zhang, Du, and Dai 2018). Thus, for k categories, the soft probability from L2R branch is defined as:

$$\sigma(\overrightarrow{Z}_{i,k}, S) = \frac{exp(\overrightarrow{Z}_{i,k}/S)}{\sum_{j=1}^{K} exp(\overrightarrow{Z}_{i,j}/S)}, \quad (9)$$

where S denotes the temperature parameter for generating soft labels. The logits of the i-th symbol of this sequence calculated by the decoder network are defined as $\overrightarrow{Z}_i = \{z_1, z_2, ..., z_K\}$. Our objective is to minimize the distance between the two branch probability distributions. Thus, the KL distance between $\overrightarrow{\mathcal{P}}_{l2r}^{'}$ and $\overleftarrow{\mathcal{P}}_{r2l}^{*}$ is computed as follows:

$$L_{KL} = S^2 \sum_{i=1}^{T} \sum_{j=1}^{K} \sigma(\overrightarrow{Z}_{i,j}, S) log \frac{\sigma(\overrightarrow{Z}_{i,j}, S)}{\sigma(\overleftarrow{Z}_{T+1-i,j}, S)}, \quad (10)$$

where S^2 ensures that the ground-truth and the probability distribution from the other branch can make comparable contributions to model training (Hinton, Vinyals, and Dean 2015), and $\overrightarrow{Z}_{i,j}$ and $\overleftarrow{Z}_{T+1-i,j}$ denote the logits from L2R and R2L branch, respectively.

Dataset	methods	ExpRate	≤1 error	≤2 error
2014	PAL	39.66	56.80	685.11
	WAP	46.55	61.16	65.21
	PGS	48.78	66.13	73.94
	PAL-v2	48.88	64.50	69.78
	DWAP-TD	49.10	64.20	67.8
	DLA	49.85	-	-
	DWAP	50.60	68.05	71.56
	DWAP-MSA	52.80	68.10	72.00
	WS WAP	53.65	-	-
	BTTR	53.96	66.02	70.28
	ABM	**56.85**	**73.73**	**81.24**
2016	PGS	36.27	-	-
	TOKYO	43.94	50.91	53.70
	WAP	44.55	57.10	61.55
	DWAP-TD	48.50	62.30	65.30
	DLA	47.34	-	-
	DWAP	47.43	60.21	63.35
	PAL-v2	49.61	64.08	70.27
	DWAP-MSA	50.10	63.80	67.40
	WS WAP	51.96	64.34	70.10
	BTTR	52.31	63.90	68.61
	ABM	52.92	**69.66**	**78.73**
2019	DWAP	47.70	59.50	63.30
	DWAP-TD	51.40	66.10	69.10
	BTTR	52.96	65.97	69.14
	ABM	**53.96**	**71.06**	**78.65**

Table 1: Comparison with prior works (in %). Note that our results are from L2R branch. The results shown in upper are partly cited from their corresponding papers.

Loss Function Specially, for the target LaTex sequence of length T, $\overrightarrow{Y}_{l2r} = \{\langle sos\rangle, Y_1, Y_2, ..., Y_T, \langle eos\rangle\}$, we denote the corresponding one-hot ground-truth label at the i-th time step as $Y_i = \{x_1, x_2, ..., x_K\}$ with $x_i \in \{0, 1\}$. The softmax probability of the k-th symbol is computed as:

$$\overrightarrow{y}_{i,k} = \frac{exp(\overrightarrow{Z}_{i,k})}{\sum_{j=1}^{K} exp(\overrightarrow{Z}_{i,j})}. \tag{11}$$

For multi-class classification, the cross-entropy losses between the target label and softmax probability for two branches are defined as:

$$L_{ce}^{l2r} = \sum_{i=1}^{T}\sum_{j=1}^{K} -Y_{i,j} log(\overrightarrow{y}_{i,j}). \tag{12}$$

$$L_{ce}^{r2l} = \sum_{i=1}^{T}\sum_{j=1}^{K} -Y_{i,j} log(\overleftarrow{y}_{T+1-i,j}). \tag{13}$$

The overall loss function is as follows:

$$L = L_{ce}^{l2r} + L_{ce}^{r2l} + \lambda L_{KL}, \tag{14}$$

where λ is a hyper-parameter to balance the recognition loss and KL divergence loss.

Methods	ExpRate	≤1 error	≤2 error	WER
Baseline	50.60	68.05	71.56	13.12
+AAM	52.64	68.62	77.25	12.12
+BML	55.23	72.58	79.08	10.25
ABM	**56.85**	**73.73**	**81.24**	**10.01**

[1] "+" means to append to baseline model.

Table 2: Ablation study (in %). We evaluate AAM and BML modules on CROHME 2014 test dataset.

Experiments

Datasets and Metrics We train our models based on the CROHME 2014 competition dataset with 111 classes of mathematical symbols and 8836 handwritten mathematical expressions, and test our model on three public test datasets: CROHME 2014, 2016, and 2019, with 986, 1147 and 1199 expressions, respectively. We use two indicators to evaluate the models: (1) Expression level: ExpRate (%), $\leq$ 1 error (%), and $\leq$ 2 error (%) represent expression recognition accuracy when zero to two structural or symbol errors can be tolerated. (2) Word level: Word Error Rate (WER(%)) (Klakow and Peters 2002) is used to evaluate such errors as substitutions, deletions, and insertions in word level.

Implementation Details Setup: Two different decoder branches in our model are set to different weight initialization methods (Glorot and Bengio 2010; He et al. 2015). For the decoder, we set n = 256, d=512, D=684 and K=113 (adding $\langle sos\rangle$ and $\langle eos\rangle$ on 111 labels). In the loss function, λ is set to 0.5. **Training:** Our proposed method is optimized with Adadelta optimizer, and its learning rate starts from 1, decaying two times smaller when the WER does not decrease within 15 epochs. And the training will stop early when the learning rate drops 10 times. We set the batch size as 16. **Testing Platform:** All the models are trained/tested on a single NVIDIA V100 16GB GPU.

Comparison with Prior Works We compare the ABM with the previous state-of-the-arts, including PAL (Wu et al. 2018), PAL-v2 (Wu et al. 2020), WAP (Zhang et al. 2017), PGS (Le, Indurkhya, and Nakagawa 2019), DWAP (WAP with DenseNet as encoder), DWAP-MSA (DWAP with multi-scale attention) (Zhang et al. 2018), DWAP-TD (DWAP with tree decoder) (Zhang et al. 2020), DLA (Le 2020), WS WAP (weakly supervised WAP) (Truong et al. 2020) and BTTR (bidirectionally trained transformer) (Zhao et al. 2021). To ensure the fairness of the performance comparison, all the methods we show do not use data augmentation. From Table 1, we can observe that: **(1)** The proposed model ABM significantly improves the recognition accuracy (ExpRate) and outperforms the baseline (DWAP) by 6.25%, 5.49% and 6.26% on three test datasets, demonstrating the success of bi-directional mutual learning module and attention aggregation module in enhancing the prediction capacity. **(2)** Compared with other methods, our ABM model is superior to the previous state-of-the-arts in terms of almost all metrics. BTTR uses a traditional transformer as a decoder, which can reduce the decoding errors of long sequences to a certain extent. However,

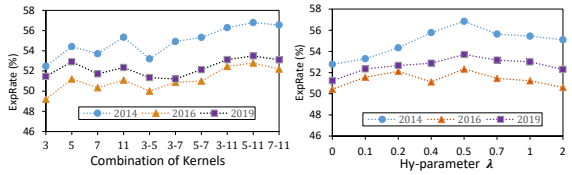

Figure 3: Parameter sensitivity analysis on combination of kernels in AAM and hy-parameter λ in Eq. (14).

the results of ≤ 1 error and ≤ 2 error show that the word error rate cannot be significantly reduced. On CROHME 2014, our model is more accurate than BTTR by large margins of 2.89%, 7.71% and 10.96% in ExpRate, ≤ 1 error and ≤ 2 error, respectively. This shows that our method can improve the performance of GRU-based models and is completely better than the transformer decoder of BTTR.

Ablation Study We conduct ablation studies to investigate the contributions of different components in the proposed network. The baseline model is a traditional encoder-decoder architecture (DWAP) (Zhang et al. 2018), which achieves 50.60% in ExpRate. From Table 2, it can be observed that: **(1)** The results of "+AAM" show that adding the attention aggregation module to the baseline model further improves the recognition performance of over 2.04%, and the multi-scale attention can benefit the baseline model because of more attentions on small symbols. **(2)** "+BML" means to equip the baseline model with an additional decoder in inverse decoding direction for mutual learning. From the results, the model can achieve 4.63% accuracy increment from 50.60% to 55.23% after taking advantage of the inverse context information. **(3)** The results of ABM show that the use of these two modules at the same time generates a cumulative effect, increasing the overall accuracy of the model by a large margin of 6.25%. To this end, we prove that every component in the proposed method can contribute to the overall recognition effectiveness.

Parameter Sensitivity We perform the sensitivity analysis on the proposed ABM on CROHME 2014 under the univariate setting. **Combination of convolution kernels in AAM:** Given that convolution kernels of different sizes have distinct receptive fields, we testify the potential values of different convolution kernels in AAM. From Fig. 3(a), the setting of multi-scale kernels can generate better performance on all datasets and the multi-scale attention is beneficial, especially for the symbols with variable scales. Our selection, i.e., combining 5*5 and 11*11, is the most robust strategy. **Hyper-parameter λ:** λ controls the trade-off between the mutual learning loss from two decoding branches and the recognition loss in Eq. (14). In Fig. 3(b), λ starts from the small factor (=0.1) to the large (=1), and the performance (ExpRate) increases at the beginning because the other branch brings valuable information from the inverse direction. However, the ExpRate tends to be lower with larger factors (like 0.7) as unstable training may happen. Thus, we set the hyper-parameter $\lambda = 0.5$.

Attention Visualization To further reveal the internal working mechanism of our proposed method, we visualize the attention process of continuous decoding, as shown in Fig. 4. Attention weights are visualized in red, and dark red denotes a higher weight in the attention map. For example, when the decoder translates the fifth character "0", viewing from left to right direction, attention can obtain historical alignment information, and from right to left direction, attention can obtain future alignment information, and eventually accurately locate the current attention position.

Methods	ExpRate	$\leq$1 error	$\leq$2 error	WER
Uni-L2R	50.60	68.05	71.56	12.75
Uni-R2L	49.24	68.05	71.56	13.26
AUM-L2R	56.24	71.56	77.34	10.76
AUM-R2L	54.71	69.94	74.12	11.01
ABM-L2R	**56.85**	**73.73**	**81.24**	**10.01**
ABM-R2L	54.86	72.01	78.90	10.86

[1] "Uni" denotes applying one branch for training.
[2] "AUM" denotes applying uni-directional mutual learning with AAM.
[3] "-L2R" and "-R2L" denotes the results generated from L2R and R2L decoder, respectively.

Table 3: Performance (in %) comparison on different decoding directions. Note that we only use one decoding branch for testing.

Methods	Prefix-2	Suffix-2	Prefix-5	Suffix-5
Uni-L2R	85.29	80.02	72.21	67.14
Uni-R2L	81.22	84.47	67.01	71.17
ABM-L2R	**88.73**	84.37	**76.35**	73.10
ABM-R2L	83.37	**87.12**	71.60	**75.86**

Table 4: Recognition accuracy (in %) of prefixes-(2, 5) (the first two or five symbols) and suffixes-(2, 5) (the last two or five symbols) on CROHME 2014 test dataset with BML.

Features Visualization Further, we visualize feature distributions of ten symbols in CROHME 2014 test set by t-SNE (Van der Maaten and Hinton 2008). We input all previous target symbols to the decoder to predict the current symbols and then visualize the features before the first full connection layer of the classifier. Fig. 5 shows that the features of different symbols generated by our method are more separable, compared to the baseline DWAP.

Evaluating Different Decoding Directions We verify the superiority of bi-directional mutual learning over the uni-directional mutual learning and original uni-trained model (Uni-L2R or Uni-R2L). We equip the Uni-directional mutual learning with AAM to form AUM for fair comparison. From Table 3, our method (ABM) improves the recognition accuracy by over 6.25%. It turns out that mutual learning between two inverse decoders is more effective.

On Long-distance Dependence One major weakness of RNNs is their unbalanced outputs with high-quality prefixes but low-quality suffixes recognition. From Table 4, there is a margin 5.27 % between prefix and suffix recognition accuracy in Uni-L2R. After using ABM, the accuracy of the L2R branch increases from 85.29% to 88.73% in prefix and from 80.02% to 84.37% in suffix. Therefore, the L2R branch can

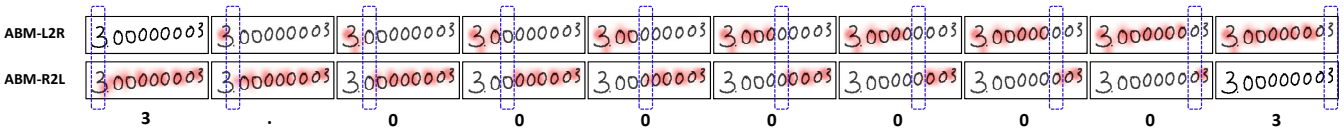

Figure 4: Coverage attention visualization process of translating handwritten mathematical expressions into LaTeX sequences in two directions (L2R and R2L). The blue box indicates the character being decoded in current time step.

Methods	ExpRate	≤1 error	≤2 error
WAP-MobileNetv2	40.61	60.32	67.59
WAP-MobileNetv2†	45.08	64.56	73.60
WAP-Xception	43.05	62.45	70.10
WAP-Xception†	46.70	68.12	75.83
DWAP-GRU	50.60	68.05	71.56
DWAP-GRU†	**56.85**	**73.73**	**81.24**
DWAP-LSTM	49.64	65.62	76.06
DWAP-LSTM†	55.13	69.95	78.84
BTTR	48.13	66.90	74.30
BTTR*	49.49	-	-
BTTR‡	51.47	69.23	76.64

[1] † denotes using AAM and BML modules.
[2] BTTR* denotes applying Bi-trained method without AJS and the results are directly cited from its paper.
[3] ‡ denotes only applying BML module as AAM is not suitable for BTTR which adopts parallel decoding.
[4] Results from L2R branch when having two branches.

Table 5: Performances (in %) of different decoders equipped with our modules.

learn the decoding knowledge from the R2L branch, and better adapt to long-distance dependence. The recognition abilities in both the directions are improved at the same time.

Generality on Different Encoders and Decoders We validate the generality of the proposed method on different encoders (MobileNetV2 (Sandler et al. 2018), Xception (Chollet 2017), and DenseNet (Huang et al. 2017)), as well as different decoders (GRU, LSTM, and Transformer). To be fair, all experiments use the same settings. As shown in Table 5, for different encoders, we replace DenseNet with MobileNetV2 and Xception, and their original modles are improved by 4.47% and 3.65%, respectively, in ExpRate. For different decoders, the GRU, LSTM, and Transformer are improved by 6.25%, 5.49%, and 3.34%, respectively, in ExpRate. We should note that the result of BTTR with one direction training is 48.13%. Therefore, our method is universal on different encoders or decoders.

Performance on Different Lengths of Expressions Further, to explore the ability of our method to decode the sequences of different lengths, we split test datasets into different groups according to the lengths of their corresponding LaTeX sequences and then compare the models for each group. Intuitively, the longer the expression, the more difficult the translation. From Table 6, BTTR solves the problem of long sequence recognition to a certain extent. This is because it uses bi-directional Transformer as a decoder.

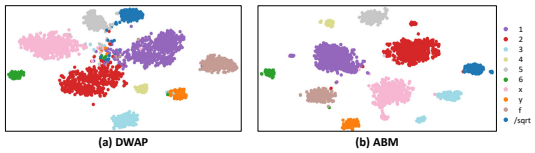

Figure 5: t-SNE Visualisation of DWAP (baseline) and ABM on CROHME 2014.

Methods	CROHME 2014				
	[1,10]	[11,20]	[21,30]	[31,40]	[41,∝]
DWAP	63.91	54.03	47.16	43.08	26.82
BTTR	68.92	58.06	50.00	52.03	20.32
ABM	**72.85**	**58.46**	**51.88**	**53.65**	**27.66**

Table 6: Accuracy for different lengths of expressions.

But for long formulas with a sequence length greater than 40, its recognition accuracy decreases instead. However, our method can perform well on any sequence length.

Statistics of Recognition Errors In order to quantitatively and accurately determine the advantages of our method, we divide the types of decoding errors into three categories: deletion error ,insertion error and substitution error. Among them, the insertion and deletion errors are mainly caused by the attention prohibition. Table 7 represents the numbers of each error in different methods.

Methods	del	ins	sub	total
WAP (Zhang et al. 2017)	202	104	252	527
DWAP (Zhang et al. 2018)	175	83	229	487
BTTR (Zhao et al. 2021)	221	56	176	453
ABM	155	49	221	425

Table 7: Statistics of three types of recognition errors.

Conclusion

We propose a novel ABM network for HMER, which uses dual-branch decoders in inverse decoding directions in a mutual learning manner. Experimental results show that the ABM is superior to the state-of-the-arts. Besides, it is applicable to existing decoders including GRU, LSTM and Transformer, and can effectively improve their performances without increasing extra parameters during inference.

Acknowledgment This work was supported by the Beijing Natural Science Foundation under Grant L191004.

References

Álvaro, F.; Sánchez, J.-A.; and Benedí, J.-M. 2016. An integrated grammar-based approach for mathematical expression recognition. *Pattern Recognition*, 51: 135–147.

Anil, R.; Pereyra, G.; Passos, A.; Ormandi, R.; Dahl, G. E.; and Hinton, G. E. 2018. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*.

Awal, A.-M.; Mouchere, H.; and Viard-Gaudin, C. 2014. A global learning approach for an online handwritten mathematical expression recognition system. *Pattern Recognition Letters*, 35: 68–77.

Bahdanau, D.; Chorowski, J.; Serdyuk, D.; Brakel, P.; and Bengio, Y. 2016. End-to-end attention-based large vocabulary speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 4945–4949. IEEE.

Chan, K.-F.; and Yeung, D.-Y. 2001. Error detection, error correction and performance evaluation in on-line mathematical expression recognition. *Pattern Recognition*, 34(8): 1671–1684.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Chollet, F. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1251–1258.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Deng, Y.; Kanervisto, A.; Ling, J.; and Rush, A. M. 2017. Image-to-markup generation with coarse-to-fine attention. In *International Conference on Machine Learning*, 980–989. PMLR.

Glorot, X.; and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256. JMLR Workshop and Conference Proceedings.

Guo, Q.; Wang, X.; Wu, Y.; Yu, Z.; Liang, D.; Hu, X.; and Luo, P. 2020. Online knowledge distillation via collaborative learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11020–11029.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708.

Kawakami, K. 2008. Supervised sequence labelling with recurrent neural networks. *Ph. D. thesis*.

Klakow, D.; and Peters, J. 2002. Testing the correlation of word error rate and perplexity. *Speech Communication*, 38(1-2): 19–28.

Lan, X.; Zhu, X.; and Gong, S. 2018. Knowledge distillation by on-the-fly native ensemble. *arXiv preprint arXiv:1806.04606*.

Lavirotte, S.; and Pottier, L. 1998. Mathematical formula recognition using graph grammar. In *Document Recognition V*, volume 3305, 44–52. International Society for Optics and Photonics.

Le, A. D. 2020. Recognizing handwritten mathematical expressions via paired dual loss attention network and printed mathematical expressions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 566–567.

Le, A. D.; Indurkhya, B.; and Nakagawa, M. 2019. Pattern generation strategies for improving recognition of handwritten mathematical expressions. *Pattern Recognition Letters*, 128: 255–262.

Le, A. D.; and Nakagawa, M. 2017. Training an end-to-end system for handwritten mathematical expression recognition by generated patterns. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, 1056–1061. IEEE.

Li, Z.; Jin, L.; Lai, S.; and Zhu, Y. 2020. Improving attention-based handwritten mathematical expression recognition with scale augmentation and drop attention. In *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 175–180. IEEE.

Luong, M.-T.; Sutskever, I.; Le, Q. V.; Vinyals, O.; and Zaremba, W. 2014. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

MacLean, S.; and Labahn, G. 2013. A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets. *International Journal on Document Analysis and Recognition (IJDAR)*, 16(2): 139–163.

MacLean, S.; and Labahn, G. 2015. A Bayesian model for recognizing handwritten mathematical expressions. *Pattern Recognition*, 48(8): 2433–2445.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–4520.

Song, G.; and Chai, W. 2018. Collaborative learning for deep neural networks. *arXiv preprint arXiv:1805.11761*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Tianfei, Z.; Jianwu, L.; Shunzhou, W.; Ran, T.; and Jianbing, S. 2020. Matnet: Motion-attentive transition network for zero-shot video object segmentation. *IEEE Transactions on Image Processing*, 29: 8326–8338.

Tianfei, Z.; Siyuan, Q.; Wenguan, W.; Jianbing, S.; and Song-Chun, Z. 2021. Cascaded parsing of human-object interaction recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Truong, T.-N.; Nguyen, C. T.; Phan, K. M.; and Nakagawa, M. 2020. Improvement of End-to-End Offline Handwritten Mathematical Expression Recognition by Weakly Supervised Learning. In *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 181–186. IEEE.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Wang, J.; Du, J.; Zhang, J.; and Wang, Z.-R. 2019. Multi-modal Attention Network for Handwritten Mathematical Expression Recognition. *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 1181–1186.

Wenguan, W.; Tianfei, Z.; Siyuan, Q.; Jianbing, S.; and Song-Chun, Z. 2021. Hierarchical human semantic parsing with comprehensive part-relation modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wu, J.-W.; Yin, F.; Zhang, Y.-M.; Zhang, X.-Y.; and Liu, C.-L. 2018. Image-to-markup generation via paired adversarial learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 18–34. Springer.

Wu, J.-W.; Yin, F.; Zhang, Y.-M.; Zhang, X.-Y.; and Liu, C.-L. 2020. Handwritten mathematical expression recognition via paired adversarial learning. *International Journal of Computer Vision*, 1–16.

Yan, Z.; Zhang, X.; Gao, L.; Yuan, K.; and Tang, Z. 2021. ConvMath: A Convolutional Sequence Network for Mathematical Expression Recognition. *2020 25th International Conference on Pattern Recognition (ICPR)*, 4566–4572.

Zhang, J.; Du, J.; and Dai, L. 2018. Multi-scale attention with dense encoder for handwritten mathematical expression recognition. In *2018 24th international conference on pattern recognition (ICPR)*, 2245–2250. IEEE.

Zhang, J.; Du, J.; Yang, Y.; Song, Y.-Z.; Wei, S.; and Dai, L. 2020. A tree-structured decoder for image-to-markup generation. In *International Conference on Machine Learning*, 11076–11085. PMLR.

Zhang, J.; Du, J.; Zhang, S.; Liu, D.; Hu, Y.; Hu, J.; Wei, S.; and Dai, L. 2017. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognition*, 71: 196–206.

Zhang, Y.; Xiang, T.; Hospedales, T. M.; and Lu, H. 2018. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4320–4328.

Zhao, W.; Gao, L.; Yan, Z.; Peng, S.; Du, L.; and Zhang, Z. 2021. Handwritten Mathematical Expression Recognition with Bidirectionally Trained Transformer. *arXiv preprint arXiv:2105.02412*.