Enhancing Math Word Problem Solving through Salient Clue Prioritization: A Joint Token-Phrase-Level Feature Integration Approach

Junyi Xie[1], Jiefeng Ma[1], Xinnan Zhang[1], Jianshu Zhang[2], Jun Du[1]*
[1] University of Science and Technology of China
[2] iFLYTEK Research
{jyxie78, jfma, zhangxinnan}@mail.ustc.edu.cn, jszhang6@iflytek.com, jundu@ustc.edu.cn

***Abstract*—Recently, deep learning-based methods, such as sequence-to-sequence and sequence-to-tree models, have shown remarkable success in Math Word Problem (MWP) solving. However, a predominant drawback of them is their insufficient attention to the inherent characteristics of math problems, including the varying significance of words within the problem. To address this issue, our paper introduces a novel approach termed "Salient Clue Prioritization (SCP)" for MWP. This approach revolves around understanding MWP from a fresh perspective by emphasizing salient clues. Our proposed approach involves two key training procedures: integrating a coarse-fine representation for multi-level comprehension during the first training phase, and dynamically adjusting attention weights based on identified keywords in the second training phase. Through extensive experimentation, our approach overcomes the limitations of existing methods by effectively identifying and adaptively utilizing salient clues to solve MWP. Our results indicate superior performance compared to other existing models, demonstrating the enhanced understanding of both the MWP itself and the relationship between words and final mathematical solutions.**

***Index Terms*—math word problem, coarse-fine adaptation, pretrained model, weighted attention**

I. Introduction

Math word problem (MWP) solving, a challenging and significant task in natural language processing (NLP), aims to convert a succinct narrative that contains numerous mathematical relationships into an equation that represents the solution.

Previous research has predominantly approached MWP solving as a sequence-to-sequence (Seq2Seq) task [1]–[5]. Recently, sequence-to-tree-structured (Seq2Tree) [6], [7] has further advanced the capabilities of MWP solvers, enabling the generation of complex mathematical expressions. The advent of pre-trained language models (PLM) such as BERT [8], has led to their application in MWP solving by researchers [9]–[12] to enhance the encoder's representation ability, resulting in significant improvements in expression generation. Moreover, [13], [14] introduce a new perspective that treats MWP as a relation extraction task.

Although using PLM is beneficial for extracting various information to generate expressions, these works overlook the semantic aspect of the corpus, and fail to consider the intrinsic features of math problems, such as the varying importance of words within the problem. To address this limitation, we propose a novel approach called "Salient Clue Prioritization (SCP)" for MWP solving. Our approach provides a new perspective on understanding MWP by highlighting salient clues, which directly impact the final answer. Our approach involves two phases. In the first phase, we utilize the Roberta [15] to extract fine-grained token-level features. Additionally, we employ a semantic segmentation method [16] to extract the coarse-grained phrase-level descriptions of the problem. By combining these two features using letter-match maps, we generate a coarse-fine integration (CFI) feature, which enhances the representation capability by incorporating features at different levels. We employ a mask-then-infer strategy, a simple yet efficient approach, to identify the salient clues. In the second phase, to enhance the model's understanding of the impact of the salient clue on the entire sentence, we dynamically adjust attention weights based on the salient clues.

Our model aims to enhance the discernment of word significance within the given problem. By effectively harnessing crucial words, our model allocates increased attention to these pivotal words while ensuring their accurate representation. This enables the model to prioritize and concentrate on words that hold substantial influence over the solution.

The contributions of this paper are as follows:

- We introduce a **coarse-fine integration** strategy for MWP solving. By simultaneously considering information from different levels, we can achieve a more comprehensive understanding of the problem.
- We propose a training model to identify the salient clue and dynamically adjust the attention weights. To

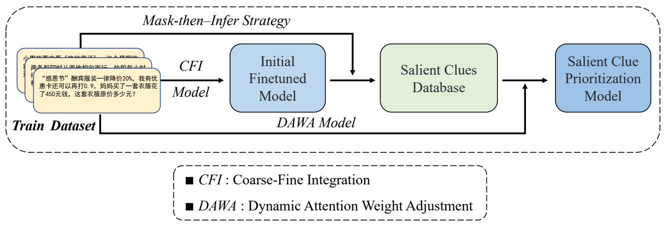
Mask-then–Infer Strategy
CFI
Model
Initial Finetuned Model
Salient Clues Database
Salient Clue Prioritization Model
Train Dataset
DAWA Model
■ CFI : Coarse-Fine Integration
■ DAWA : Dynamic Attention Weight Adjustment

Fig. 1. Overall system pipeline. Firstly, the initial finetuned model is obtained through coarse-fine integration (CFI). Secondly, the mask-then-infer strategy is employed to create a salient clue database. Dynamic training with attention weight adjustment (DAWA) is performed with the salient clues.

*corresponding author

the best of our knowledge, we are the first to propose **salient clue prioritization** for MWPs, which is of great significance to emphasize the keywords that are decisive to the answers.

- Large experiments show that the proposed model is efficient and outperforms state-of-the-art baselines on MWP solvers with the similar encoder-decoder framework.

II. Related Work

A. Math Word Problem Solving

Math word problem solving can be firstly dated back to the 1960s [17]. In the past, many methods relied on human-designed rules [18] or statistical learning approaches [19]. These methods concentrate on feature designing and expression template construction. Recently, deep learning-based models show great potential for solving MWPs. [1] proposed the commonly used dataset Math23K and first introduce a vanilla Seq2Seq model. Following it, [3] combines reinforcement learning to improve performance. Subsequently, semantic-parsing [2], template-based [20] methods, and group attention mechanism [5] are incorporated to enhance performance. Then, a new model using Seq2Tree is proposed by [7], which introduces a goal-driven tree-structured (GTS) model to generate the expression tree step by step. Following the tree-structured decoder, [21] uses multi-encoders and multi-decoders to strengthen the generation ability. [22] introduces syntactic dependency to enhance the understanding of math word problems. [23], [24] proves that commonsense knowledge can also improve the understanding performance. [16], [25], [26] use graph neural networks to obtain useful information from the problems. Furthermore, [27] introduces a situation model for algebra story problems. [28] applies contrast learning to improve the encoder. [29] proposes many auxiliary tasks to enhance the symbolic reasoning ability. [30] improves MWPs with pre-trained knowledge and hierarchical reasoning. Furthermore, [31] proposes a linguistic logic-enhanced framework for generating expression trees and their corresponding interpretation. Recently, some researchers try other perspectives to solve MWPs. [32] applies pointer transformer and [13] treats the equation as a directed cyclic graph to obtain the expression. [14] introduces a new decoder treating it as an iterative relation extraction task. [33] proposes the M-tree codes and a sequence-to-code (Seq2Code) framework. [34] employs multiple consistent reasoning views to address the challenges of MWP.

B. Attention Mechanism

Attention mechanisms have gained significant attention in the field of NLP due to their ability to capture the contextual dependencies and relevance of different parts within a sequence. It is initially proposed by [35], where it was employed in improving the translation quality. Inspired by the success of attention mechanisms, [36] design a hierarchical attention network that leverages attention to enhance the understanding of important words. The advent of self-attention can be attributed to the Transformer [37], which enables the model to simultaneously attend to information from various positions and representation subspace. This is achieved by computing a weighted average of feature representations, where the weights are determined by the similarity scores between pairs of representations.

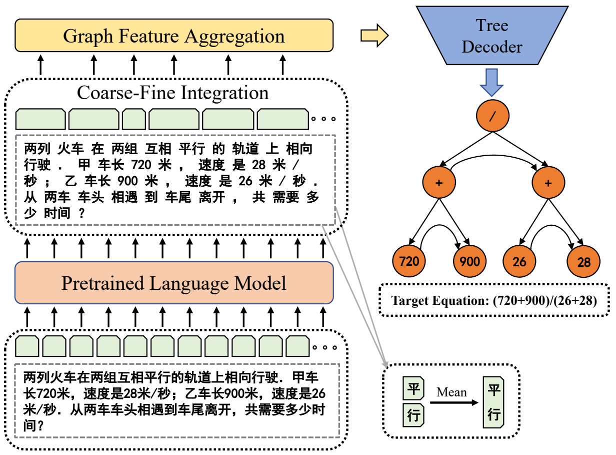
Graph Feature Aggregation
Tree Decoder
Coarse-Fine Integration
两列 火车 在 两组 互相 平行 的 轨道 上 相向 行驶 . 甲 车长 720 米 , 速度 是 28 米 / 秒 ; 乙 车长 900 米 , 速度 是 26 米 / 秒 . 从 两车 车头 相遇 到 车尾 离开 , 共 需要 多 少 时间 ?
/
+
+
720
900
26
28
Target Equation: (720+900)/(26+28)
Pretrained Language Model
两列火车在两组互相平行的轨道上相向行驶. 甲车长720米, 速度是28米/秒; 乙车长900米, 速度是26米/秒. 从两车车头相遇到车尾离开, 共需要多少时间?
平
行
Mean
平
行

Fig. 2. First round model training. First, a PLM is used to process the problem, obtaining fine-level representations. Subsequently, a mean operation is applied to the creating of coarse-fine integrated representations. After incorporating the graph features, a tree decoder is employed to generate the target equation.

III. Proposed Method

In the context of math word problems, we denote the given problem as P, which consists of a sequence of word tokens and numeric values. Let $\mathcal{V}_P = \{v_1, ..., v_m\}$ represent the set of word tokens in P, and let $\mathcal{N}_P = \{n_1, ..., n_l\}$ denote the set of quantities in P, respectively. The problem of our interest is to map P to a correct mathematical expression E_P, which can be used to compute the answer using a series of primitive mathematical operations $\mathcal{V}_{\text{OP}} = \{+, -, \times, /\}$.

In practical math word problem solving, predefined constants such as π and 1 are often required. To account for these constants, we define the constant set $\mathcal{V}_{\text{C}} = \{1, \pi\}$. Therefore, the goal is to maximize the probability $P(E_P|\mathcal{V}_P \cup \mathcal{V}_{\text{OP}} \cup \mathcal{V}_c \cup \mathcal{N}_P)$, which represents the probability of obtaining the correct mathematical expression E_P given the sets of word tokens, operators, constants, and quantities present in the problem.

A. Overview

Our model framework is illustrated in Figure 1. We preprocess our quantity by replacing them with a general quantity token "*<NUM>*" before encoding. Then, we obtain fine-grained token-level features and coarse-grained phrase-level descriptions of the problem through different processes. By combining these two features using letter-match maps, we create the CFI feature, which enhances the representation capability by integrating features at different levels. This CFI feature is then fed into a graph transformer with two constructed graphs [16] to learn a graph representation. Subsequently, the graph representation is input into a tree-structured decoder [7] to generate the final mathematical expression.

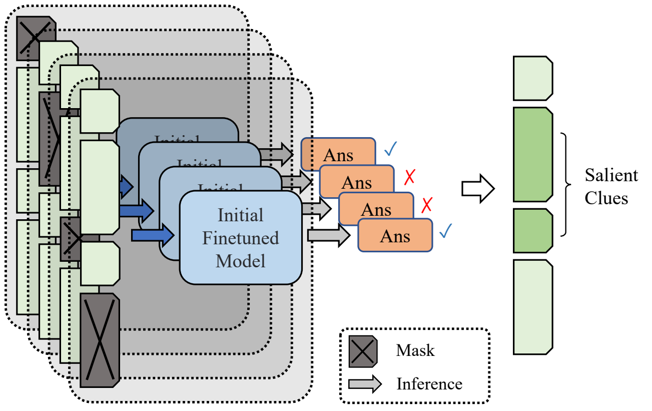

Fig. 3. Mask-then-infer strategy. We first iteratively mask phrases in a sentence, resulting in a modified sentence then fed into the finetuned model for inference. If the answer changes from correct to incorrect, it indicates that the masked phrase is crucial to the final answer, thus presuming it is a salient clue.

To gain a better understanding of the importance of different words, we design a simple but efficient way, referred to as mask-then-infer, to identify the salient clues in the problem. This enables us to build a salient clue database $\mathcal{D}_{\text{key}}$. To fully utilize the significance of salient clue, we load the best-performing model from previous training and perform continuous training while incorporating a dynamic attention weight adjustment in the self-attention layer [8].

B. Coarse-Fine Integration Representation

In order to enhance the MWP solver's understanding of problems, it is beneficial to have a multi-granularity representation during the encoder-decoder procedure. Therefore, as shown in Figure 2, we design a letter-match map to integrate the fine-grained representation obtained from the pre-trained language models such as BERT or Roberta, along with the coarse-grained representation from Jieba[1] segmentation.

Considering that PLM could provide a strong foundation with fine-grained features, we employ Roberta to capture token-level representations of the text. Additionally, to further comprehend the text itself, we utilize the coarse-grained representation to grasp a better understanding of the problem by dividing it into meaningful and independent phrases, facilitating subsequent text analysis and processing. To effectively integrate the strengths of both approaches and achieve a more comprehensive and accurate representation of the math problem, we propose a coarse-fine granularity relation graph for their fusion.

We denote the fine-grained tokens and their features for problem P obtained from Roberta as $\{t_1, t_2, ..., t_M\}$ and $\{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_M\}$, respectively, where M represents the length of sequence encoded by Roberta. The coarse-grained phrase-level description is denoted as $\mathcal{C}_P = \{\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_N\}$, where N is the number of segmented phrases. To unify the features from different granularities, we introduce a relation graph $G \in \mathbb{R}^{M\times N}$ to align the fine and coarse granularities. The relation graph is defined as follows:

$$G_{ij} = \begin{cases} 1 & \text{if } t_i \text{ in } \tilde{p}_j \\ 0 & \text{otherwise} \end{cases}. \tag{1}$$

Once we have obtained the relation graph G that captures the relationship between features from different levels, we are capable of getting the CFI feature as follows:

$$\mathbf{h}_j = \sum_{\{i|G_{ij}=1\}} \mathbf{f}_i / \sum_{i=1}^{M} G_{ij}, \quad j = 1, ..., N. \tag{2}$$

Following the above producer, we obtain the CFI representation matrix $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_N]$, which then is processed by graph convolution network [38] with two constructed graphs [16].

C. Salient Clue Retrieval

The salient clue is crucial in problem-solving as it directly influences how the model understands the problem, thereby impacting the answer. Therefore, we propose a simple yet efficient approach, referred to as the mask-then-inference strategy, to identify salient clues in the math problem.

After training the model using the CFI feature, we obtain the best-performing model from the first round of training. During the inference phase on the training dataset, as illustrated in Figure 3, we employ a masking technique, where we mask tokens of each phrase as "<*unk*>" one by one in the problem. Subsequently, we evaluate the correctness of the generated results.

Specifically, we consider a phrase as a salient clue if the answer changes from correct to incorrect or from incorrect to correct after masking it. Conversely, if the answer remains unchanged, we assume that the phrase has little impact on the final result. Therefore, we can retrieve the phrases that significantly influence the final result.

D. Attention Weight Adjustment

Following the dot-product attention in Transformer [37], which is given as

$$\text{Attention}\,(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\boldsymbol{V}, \tag{3}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N' \times d}$ with N' being the sequence length, and d being the hidden dimension.

To emphasize the importance of salient clues in a sentence, we use attention weight adjustment by introducing a weight vector $\mathbf{w} \in \mathbb{R}^{N' \times 1}$, which is illustrated in Figure 4. The weight vector $\mathbf{w}$ is defined as follows:

$$\mathbf{w}_i = \begin{cases} \lambda & \text{if } i\text{-th phrase is a salient clue} \\ 1 & \text{otherwise} \end{cases}, \tag{4}$$

where λ is a learned parameter. Based on this, we design the adjusted attention mechanism as follows:

$$\text{Attention}\,(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \mathbf{W}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top \odot \mathbf{W}}{\sqrt{d}}\right)\boldsymbol{V}, \tag{5}$$

where $\odot$ denotes the element-wise product, and $\mathbf{W} = \mathbf{w}\mathbf{w}^\top$ represents the weight correlation matrix.

[1]http://github.com/fxsjy/jieba

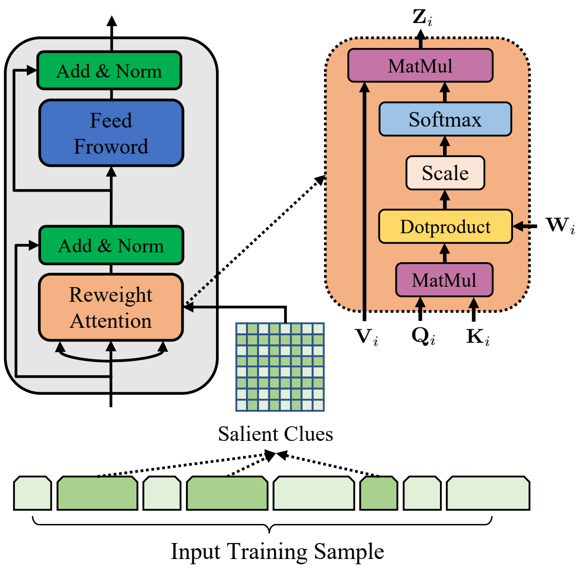

Fig. 4. Dynamic attention weight adjustment. The attention mechanism is re-weighted to dynamically adjust the attention based on the identified salient clues.

TABLE I
THE ACCURACY COMPARISON TO DIFFERENT METHODS. VAL ACC.: ACCURACY OF THE VALUE CALCULATED FROM THE EQUATIONS; ♢: USING THE SAME SEQ2TREE FRAMEWORK; ♣: USING THE PRE-TRAINED LANGUAGE MODELS.

Framework	Model	Encoder	Val Acc. (%)
Seq2Seq	Math-EN [39]	RNN	66.7
Seq2Seq	T-RNN [20]	RNN	66.9
Seq2Seq	Group-ATT [5]	RNN	69.5
Seq2Tree♢	GTS [20]	RNN	75.6
Seq2Tree♢	Graph2Tree [16]	RNN	77.4
Seq2Tree♢	BERT-Tree [40]	PLM♣	83.9
I-RE	Re-Deduction [14]	PLM♣	85.1
Seq2Code	SUMC-Solver [33]	RNN	77.4
Seq2Code	SUMC-Solver [33]	PLM♣	82.5
Seq2Tree♢	Logic-Solver [31]	PLM♣	83.4
Seq2Tree♢	Ours	PLM♣	**86.9**

IV. EXPERIMENTS

A. Datasets and Experiment Setup

1) Datasets: We conduct experiments on Math23k [1], a Chinese math word problem dataset, which contains 23,161 problems labeled with structured equations and answers. The dataset encompasses 822k words, and 70.1k sentences with 2,187 templates, which is one of the most challenging and popular Chinese MWPs datasets.

2) Experiment Setup: We implement our model with PyTorch and conduct experiments with an NVIDIA-Tesla M40 24GB GPU. We replace all numbers in the problem text with "*<NUM>*". The sizes of word embeddings and all hidden states for other layers are all set as 512. The batch size here we used is 32. We optimize Roberta with ADAMW [41] optimizer and use ADAM [42] for the decoder. The initial fine-tuning learning rate is set as $5e^{-5}$ and $1e^{-3}$ for pre-trained BERT models and tree-decoder. We set the dropout rate as 0.3 and weight decay as $1e^{-5}$ to avoid over-fitting.

TABLE II
ABLATION STUDIES OF THE DIFFERENT COMBINATIONS ON THE MATH23K DATASET. ALL MODELS USE A TREE DECODER. CFI: COARSE-FINE INTEGRATION; DAWA: DYNAMIC ATTENTION WEIGHT ADJUSTMENT.

Encoder	Graph	CFI	DAWA	Val Acc.
RNN-based	-	-	-	75.6
RNN-based	√	-	-	77.4
Bert	-	-	-	82.8
Roberta	-	-	-	83.6
Roberta	-	√	-	84.9
Roberta	√	√	-	85.8
Roberta	√	√	√	**86.9**

TABLE III
COMPARISON OF THE PROPOSED SALIENT CLUES STRATEGY FOR DIFFERENT INITIALIZED ATTENTION WEIGHTS ON THE MATH23K DATASET.

Initial Value	Stable Value	Val Acc. (%)
1.0	1.192	86.6
1.1	1.194	**86.9**
1.2	1.190	86.1
1.3	1.203	86.0

B. Overall Result

1) Baselines.: We classify the baseline approaches into the following groups: Seq2Seq, Seq2Tree, Seq2Code and iterative extraction of relationships (I-RE). We compare our model to an extensive set of baselines and state-of-the-art models: **Math-EN** [39] leverages equation normalization techniques to reduce the target space. **T-RNN** [20] employs recursive neural networks for predicted tree-structured templates. **GROUP-ATT** [5] adopts the concept of multi-head attention from the Transformer architecture [37]. **GTS** [20] develops goal-driven tree-structured neural networks to generate expression trees. **Graph2tree** [16] follows GTS using a tree-decoder while merging the graph transformer to enhance the basic relations between different quantities. **BERT-Tree** [40] applies BERT as an encoder but fusing constructive-learning approach to better understand MWP patterns and perceive the divergence of patterns. **Re-Deduction** [14] uses pre-trained language models as encoders while formulating MWP solving as a complex relation extraction task. **LogicSolver** [31] incorporate mathematical logic knowledge through logical prompt-enhanced learning. **SUMC-Solver** [33] analyze mathematical expressions with M-tree codes and Seq2Code.

2) Performances: The main results on Math23K are shown in Table I. For the Math23K dataset, we notice that our model outperforms all baselines on this Chinese dataset. Even if we apply a similar tree-decoder like GTS, Graph2Tree, BERT-tree, and Logic-Solver, our model achieves at least 4.1% higher than others. Compared with other methods which also use the pre-trained model such as Re-Deduction and SUMC-Solver, we still outperform them by nearly 1.4%. It is noteworthy that our model demonstrates superior performance when applied to

Case1: 妈妈 买 了 4 千克 香蕉 和 3.5 千克 苹果， 一共 花 了 51.2 元． 已知 香蕉 每千克 7.2 元， 苹果 每千克 多少 元？

Mom bought 4 kilograms of bananas and 3.5 kilograms of apples, spending a total of 51.2 yuan. It is known that the price of bananas is 7.2 yuan per kilogram. What is the price of apples per kilogram?

Graph2Tree	$(51.2 - 7.2 \times 3.5)/4$	**Ours**	$(51.2 - 7.2 \times 4)/3.5$

Case2: 轮船 从甲港 开往 乙港共行 3 天， 第一天 行 了 全程 的 (2/5)， 第二天 行 了 全程 的 (1/3)， 第三天 行 了 280km， 甲港 到 乙港 的 航程 有 多少 km？

A ship travels from Port A to Port B over the course of 3 days. On the first day, it covers 2/5 of the total distance. On the second day, it covers 1/3 of the total distance. On the third day, it covers 280 kilometers. How long is the journey from Port A to Port B?

Graph2Tree	$280/(1 - 2/5 - 1/3)\times 2/5$	**Ours**	$280/(1 - 2/5 - 1/3)$

Case3: 学校 要 植树 500 棵， 把 植树 任务 按 2： 3 分给 5 、 6 两个 年级． 实际 植树 时， 六年级 超过 原 分配任务 的 (2/5)， 六年级 实际 植树 多少 棵？

The school needs to plant 500 trees and assigns the tree-planting task in a 2:3 ratio to grades 5 and 6. During the actual tree-planting process, grade 6 exceeded its originally assigned task by 2/5. How many trees did grade 6 actually plant?

Graph2Tree	$500 \times 3/(2 + 3 + 3) \times (1 + 2/5)$	**Ours**	$500 \times 3/(2 + 3) \times (1 + 2/5)$

Fig. 5. Three examples of solving MWPs with our strategy and Graph2Tree. We can notice that our model outperforms Graph2Tree and have a better and more accurate understanding of the MWPs.

Chinese datasets, primarily due to its semantic segmentation commonly more explicit at the phrase level.

C. *The Ablation Study and Parameter Analysis*

To investigate the impact of different components and hyperparameters in our model, we perform comprehensive ablation studies and conduct a detailed parameter analysis on the Math23K dataset.

1) The Ablation Study: In order to investigate how well our model performs as compared to state-of-the-art models using explicit tree-decoder and PLM, we do some ablation studies shown in Table II, and analyze the effect of each component. **Effect of Graph Transformer.** The accuracy achieved by the GTS with the RNN-based encoder and tree-decoder is 75.6%. By incorporating the graph transformer based on this model, the accuracy significantly improves to 77.4%. **Effect of PLM.** Inspired by the BERT-Tree [40], we replace the encoder with a PLM. Compared to the RNN base, we use BERT as the encoder, resulting in an improvement of approximately 5.4%. We further experiment with Roberta as the encoder, leading to a remarkable accuracy of 83.6%. **Effect of Coarse-Fine Integration.** By incorporating the CFI feature into the fixed encoder-decoder basic structure, it resulted in an impressive accuracy performance of 84.9%, surpassing all models that followed the PLM-based encoder and tree-decoder framework. **Effect of Attention Weight Adjustment.** The inclusion of salient clue identification and dynamic attention weight adjustment in our model yielded a substantial accuracy improvement, with a performance of 86.8%. This notable gain provides compelling evidence for the significance and efficiency of the attention weight adjustment technique.

2) Parameter Analysis: To investigate the influence of the initial value of attention weights on the performance, we conduct an experiment by varying the initial attention weight from 1 to 1.3 with an increment of 0.1. The results, as shown in Table III, reveal that the attention weights exhibit a notable trend towards 1.2 across different initial values. Moreover, we observe that the best performance is achieved when the initial value is set to 1.1.

D. *Case Study*

Subsequently, we perform a detailed case analysis, presenting three specific cases as depicted in Figure 5. By leveraging our salient clue prioritization strategy, our model demonstrates notable improvements over existing methods in various aspects. Firstly, it exhibits enhanced accuracy in predicting operations, constants, and number words, outperforming the Graph2Tree model which tends to generate erroneous expressions. Moreover, our model displays outstanding capabilities in extracting and generating precise expressions. This ability significantly contributes to a better understanding and analysis of the decision-making clues, as well as facilitates comprehension of the key factors guiding the model's predictions.

In summary, our joint token-phrase-level feature integration approach empowers our model with enhanced accuracy and attention prioritization in solving MWPs. This analysis highlights the superiority and effectiveness of our proposed approach, providing promising insights into the advancement of MWP-solving techniques.

V. Conclusion and Future Work

We introduce a novel approach termed "Salient Clue Prioritization (SCP)" for MWPs, which enhances MWPs through salient clue prioritization. We perform extensive experiments to evaluate our models against other baseline models, and ablation studies further prove the efficiency of our model components. Results show that our joint token-phrase-level feature integration approach outperforms other baselines on the MWP task. In future work, we aim to further explore how to provide better explanations of the entire problem-solving process. Our goal is to transform the problem-solving process into a human-like reasoning process, allowing us to understand and follow the step-by-step reasoning involved in finding a solution.

References

[1] Y. Wang, X. Liu, and S. Shi, "Deep neural solver for math word problems," in *Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP)*, 2017, pp. 845–854.

[2] D. Huang, S. Shi, C.-Y. Lin, and J. Yin, "Learning fine-grained expressions to solve math word problems," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017, pp. 805–814.

[3] D. Huang, J. Liu, C.-Y. Lin, and J. Yin, "Neural math word problem solver with reinforcement learning," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 213–223.

[4] T.-R. Chiang and Y.-N. Chen, "Semantically-aligned equation generation for solving and reasoning math word problems," *arXiv preprint arXiv:1811.00720*, 2018.

[5] J. Li, L. Wang, J. Zhang, Y. Wang, B. T. Dai, and D. Zhang, "Modeling intra-relation in math word problems with different functional multi-head attentions," in *Proceedings of the 57th annual meeting of the association for computational linguistics*, 2019, pp. 6162–6167.

[6] Q. Liu, W. Guan, S. Li, and D. Kawahara, "Tree-structured decoding for solving math word problems," in *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, 2019, pp. 2370–2379.

[7] Z. Xie and S. Sun, "A goal-driven tree-structured neural model for math word problems." in *IJCAI*, 2019, pp. 5299–5305.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[9] Y. Lan, L. Wang, Q. Zhang, Y. Lan, B. T. Dai, Y. Wang, D. Zhang, and E.-P. Lim, "Mwptoolkit: An open-source framework for deep learning-based math word problem solvers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 11, 2022, pp. 13 188–13 190.

[10] M. Tan, L. Wang, L. Jiang, and J. Jiang, "Investigating math word problems using pretrained multilingual language models," *arXiv preprint arXiv:2105.08928*, 2021.

[11] Z. Liang, J. Zhang, J. Shao, and X. Zhang, "Mwp-bert: A strong baseline for math word problems," 2021.

[12] J. Shen, Y. Yin, L. Li, L. Shang, X. Jiang, M. Zhang, and Q. Liu, "Generate & rank: A multi-task framework for math word problems," *arXiv preprint arXiv:2109.03034*, 2021.

[13] Y. Cao, F. Hong, H. Li, and P. Luo, "A bottom-up dag structure extraction model for math word problems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, 2021, pp. 39–46.

[14] Z. Jie, J. Li, and W. Lu, "Learning to reason deductively: Math word problem solving as complex relation extraction," *arXiv preprint arXiv:2203.10316*, 2022.

[15] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[16] J. Zhang, L. Wang, R. K.-W. Lee, Y. Bin, Y. Wang, J. Shao, and E.-P. Lim, "Graph-to-tree learning for solving math word problems," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3928–3937.

[17] E. Charniak, "Computer solution of calculus word problems," in *Proceedings of the 1st international joint conference on Artificial intelligence*, 1969, pp. 303–316.

[18] Y. Bakman, "Robust understanding of word problems with extraneous information," *arXiv preprint math/0701393*, 2007.

[19] R. Koncel-Kedziorski, H. Hajishirzi, A. Sabharwal, O. Etzioni, and S. D. Ang, "Parsing algebraic word problems into equations," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 585–597, 2015.

[20] L. Wang, D. Zhang, J. Zhang, X. Xu, L. Gao, B. T. Dai, and H. T. Shen, "Template-based math word problem solvers with recursive neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 7144–7151.

[21] Y. Shen and C. Jin, "Solving math word problems with multi-encoders and multi-decoders," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 2924–2934.

[22] X. Lin, Z. Huang, H. Zhao, E. Chen, Q. Liu, H. Wang, and S. Wang, "Hms: A hierarchical solver with dependency-enhanced understanding for math word problem," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 5, 2021, pp. 4232–4240.

[23] Q. Wu, Q. Zhang, J. Fu, and X.-J. Huang, "A knowledge-aware sequence-to-tree network for math word problem solving," in *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, 2020, pp. 7137–7146.

[24] Q. Wu, Q. Zhang, Z. Wei, and X.-J. Huang, "Math word problem solving with explicit numerical values," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pp. 5859–5869.

[25] S. Li, L. Wu, S. Feng, F. Xu, F. Xu, and S. Zhong, "Graph-to-tree neural networks for learning structured input-output translation with applications to semantic parsing and math word problem," *arXiv preprint arXiv:2004.13781*, 2020.

[26] Y. Zhang, G. Zhou, Z. Xie, and J. X. Huang, "Hgen: Learning hierarchical heterogeneous graph encoding for math word problem solving," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 816–828, 2022.

[27] Y. Hong, Q. Li, R. Gong, D. Ciao, S. Huang, and S.-C. Zhu, "Smart: A situation model for algebra story problems via attributed grammar," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 14, 2021, pp. 13 009–13 017.

[28] Z. Liang and X. Zhang, "Solving math word problems with teacher supervision." in *IJCAI*, 2021, pp. 3522–3528.

[29] J. Qin, X. Liang, Y. Hong, J. Tang, and L. Lin, "Neural-symbolic solver for math word problems with auxiliary tasks," *arXiv preprint arXiv:2107.01431*, 2021.

[30] W. Yu, Y. Wen, F. Zheng, and N. Xiao, "Improving math word problems with pre-trained knowledge and hierarchical reasoning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021, pp. 3384–3394.

[31] Z. Yang, J. Qin, J. Chen, L. Lin, and X. Liang, "Logicsolver: Towards interpretable math word problem solving with logical prompt-enhanced learning," *arXiv preprint arXiv:2205.08232*, 2022.

[32] B. Kim, K. S. Ki, D. Lee, and G. Gweon, "Point to the expression: Solving algebraic word problems using the expression-pointer transformer model," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 3768–3779.

[33] B. Wang, J. Ju, Y. Fan, X.-Y. Dai, S. Huang, and J. Chen, "Structure-unified m-tree coding solver for mathword problem," *arXiv preprint arXiv:2210.12432*, 2022.

[34] W. Zhang, Y. Shen, Y. Ma, X. Cheng, Z. Tan, Q. Nong, and W. Lu, "Multi-view reasoning: Consistent contrastive learning for math word problem," *arXiv preprint arXiv:2210.11694*, 2022.

[35] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[36] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1480–1489.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[38] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[39] L. Wang, Y. Wang, D. Cai, D. Zhang, and X. Liu, "Translating a math word problem to an expression tree," *arXiv preprint arXiv:1811.05632*, 2018.

[40] Z. Li, W. Zhang, C. Yan, Q. Zhou, C. Li, H. Liu, and Y. Cao, "Seeking patterns, not just memorizing procedures: Contrastive learning for solving math word problems," *arXiv preprint arXiv:2110.08464*, 2021.

[41] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.