Machine Intelligence Research
www.mi-research.net

19(5), October 2022, 425-438
DOI: 10.1007/s11633-022-1351-2

# Clause-level Relationship-aware Math Word Problems Solver

Chang-Yang Wu[1] Xin Lin[1] Zhen-Ya Huang[1] Yu Yin[1]
Jia-Yu Liu[1] Qi Liu[1,2] Gang Zhou[3]

[1] Anhui Province Key Laboratory of Big Data Analysis and Application, School of Data Science, University of Science and Technology of China, Hefei 230026, China

[2] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230088, China

[3] Laboratory of Mathematical Engineering and Advanced Computing, Information Engineering University, Zhengzhou 450001, China

**Abstract:** Automatically solving math word problems, which involves comprehension, cognition, and reasoning, is a crucial issue in artificial intelligence research. Existing math word problem solvers mainly work on word-level relationship extraction and the generation of expression solutions while lacking consideration of the clause-level relationship. To this end, inspired by the theory of two levels of process in comprehension, we propose a novel clause-level relationship-aware math solver (CLRSolver) to mimic the process of human comprehension from lower level to higher level. Specifically, in the lower-level processes, we split problems into clauses according to their natural division and learn their semantics. In the higher-level processes, following human's multi-view understanding of clause-level relationships, we first apply a CNN-based module to learn the dependency relationships between clauses from word relevance in a local view. Then, we propose two novel relationship-aware mechanisms to learn dependency relationships from the clause semantics in a global view. Next, we enhance the representation of clauses based on the learned clause-level dependency relationships. In expression generation, we develop a tree-based decoder to generate the mathematical expression. We conduct extensive experiments on two datasets, where the results demonstrate the superiority of our framework.

**Keywords:** Artificial intelligence (AI), artificial neural network (ANN), computational mathematics, machine intelligence, machine learning.

**Citation:** C. Y. Wu, X. Lin, Z. Y. Huang, Y. Yin, J. Y. Liu, Q. Liu, G. Zhou. Clause-level relationship-aware math word problems solver. *Machine Intelligence Research*, vol.19, no.5, pp.425–438, 2022. http://doi.org/10.1007/s11633-022-1351-2

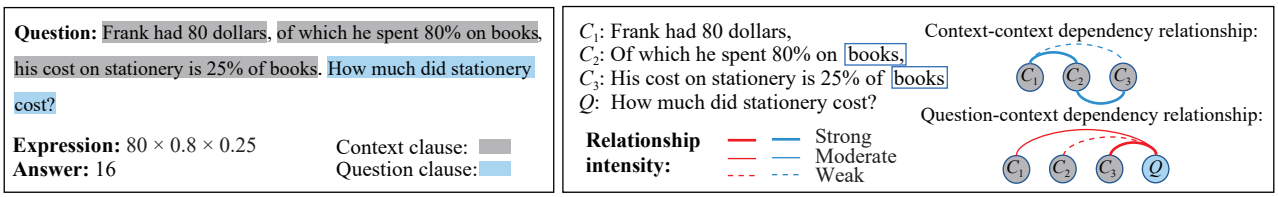
## 1 Introduction

Automatically solving math word problems (MWP) aims to answer problems with mathematical expressions according to their textual description. It is an essential task for artificial intelligence (AI) research as it requires computers to understand how humans comprehend natural language[1], apply mathematical rules, use the domain knowledge and conduct logical reasoning[2]. In addition, it is of great significance to serve as a good testbed to evaluate the internal abilities of machines[3]. Generally, a typical MWP is a short narrative that describes a problem and poses a question on an unknown numeric variable.

A toy example of MWP is illustrated in Fig. 1(a); the problem begins with some contextual information ("Frank had ⋯") and ends with a posed question ("how much ⋯ ?"). To solve this problem, one should comprehend the narrative and convert it into a mathematical expression that is composed of operators (e.g., ×, +) and quantities (e.g., 80, 0.8).

In the literature, many efforts have been made to solve math word problems, and good performance has been achieved. Previous studies have mainly focused on expression generation[4–6], quantity relationships or word-level semantic relationships[7–9] while ignoring the understanding of clause-level relationships in the comprehension processes. Therefore, such models are generally far from enough for understanding math word problems as deeply as humans. The theory of human comprehension[10, 11] reflects the process of human understanding from a lower level to a higher level. At the lower level, humans first recognize words and their roles in a clause (a short sentence, e.g., "of which he spent 80% on books") and understand semantic information from words (e.g., "spent", "80%", "on") to clause (e.g., "of which

Research Article
Special Issue on Brain-inspired Machine Learning
Manuscript received April 11, 2022; accepted June 17, 2022; published online August 25, 2022
Recommended by Associate Editor Cheng-Lin Liu
Colored figures are available in the online version at https://link.springer.com/journal/11633




(a) Context and question clause in math word problem (b) Clause-level dependency relationships

Fig. 1 An example of math word problems and clause-level dependency relationships in the problem

he ⋯ on books"). Then, at the higher level, humans establish dependency relationships between clauses (e.g., "his cost on stationery is 25% of books" is strongly connected with "of which he spent 80% on books"), which is a clause-level relationship. To this end, we hope to capture and exploit the clause-level relationship to improve the MWP solver imitating humans.

However, there are several technical challenges along this line. First, when considering the relationship between two clauses, human beings are influenced not only by the overall semantic of clauses, but also some dominant words in the clauses[12]. For example in Fig. 1(b), obviously, the dependency relationship between $C_1$ ("Frank had ⋯") and $C_2$ ("of which ⋯") is stronger than the relationship between $C_1$ and $C_3$ from the perspective of overall semantic information. In addition, the word "books" in the clauses $C_2$ and $C_3$ has a significant effect on the relationship between them. Thus, to understand the relationship of clauses, solvers need to preserve whole semantics of the clauses as well as consult words relevance. Second, humans distinguish clauses into the context clause and question clause in reading as shown in Fig. 1, which take different roles in problem comprehension. The dependency relationships among context clauses are independent of the question clause. The question clause determines the goal of the problem; thus the relationships between the question and context clauses may vary with the semantics of the question. For example, in Fig. 1(b), the relationship between context clauses (context-context dependency relationships in Fig. 1(b)) is independent of question clause $Q$ ("How much ⋯"), and clause $Q$ plays a decisive role in the question-context dependency relationship. Although humans can capture these two relationships without much effort, machines may struggle with them.

To address the challenges above, we model the dependency relationships between clauses from both local and global views, inspired by the local and global strategies[11] of human comprehension. We propose a novel clause-level relationship-aware solver (CLRSolver) to model the dependency relationships between the clauses. Following human understanding from lower to higher, we first learn the clause semantics in the lower level, and propose a multi-view method to capture the clause-level relationships. Specifically, in the local view, we develop a CNN-based module to learn relationships between clauses from word relevance. In the global view, we design two novel relationship-aware mechanisms for learning context-context and question-context dependency relationships based on the overall semantics of clauses. Next, we enhance the clause representations based on the learned relationships, and apply a tree-based decoder following the hierarchical math solver (HMS)[13] framework to generate expressions with enhanced clause representations. We conduct extensive experiments on two real-world datasets. Experimental results demonstrate that CLRSolver not only achieves a better performance of solution prediction but also shows the ability to capture the dependency relationships between clauses.

## 2 Related work

### 2.1 Math word problems

Research on solving MWP dates back to several decades ago[14, 15]. MWP solvers can be roughly divided into four categories according to the technology paradigms they adopted: rule-based methods[15–18], statistical machine learning methods[7, 19, 20], semantic parsing methods[4, 21, 22] and deep learning methods[3, 5, 9, 13, 23–30]. Specifically, rule-based methods[17, 31] manually craft rules and schemas for pattern matching. Therefore, these solvers have the limitations of generalization and human intervention. Statistical machine learning methods[7, 19] use traditional machine learning methods to solve problems with the help of predefined expression templates. Semantic parsing methods, such as [4, 22], map the sentences from problem statements into structured logic representations and then generate the answer with quantitative reasoning. Such methods still have visible limitations for the indispensability of tremendous human effort on feature engineering and deficient generality.

Deep learning methods have been widely adopted for their advantages of automatic feature learning and effective generality. With the help of deep learning, researchers have developed several approaches, such as Seq2Seq[24], Seq2Tree[5], Seq2DAG[32], Graph2Tree[9] and reinforcement learning[25]. For example, Wang et al.[24] applied a seq2seq model to translate textual description to a solution expression. Inspired by the goal-driven mechanism in the problem-solving of humans, Xie and Sun[5] proposed a tree-structured decoder to generate mathematic-

al expressions in a tree-like manner. To release the merits of the relationships and order information among the quantities, Zhang et al.[9] applied two graphs to represent the relationships and order information among the quantities in MWP. To incorporate common-sense knowledge of global expression information, Wu et al.[29] built an entity graph for each problem to obtain knowledge-aware problem understanding. Lin et al.[13] proposed an HMS model that imitated human word-clause-problem reading habits. Wu et al.[30] took the edge label information between words and the long-range word relationship into consideration and proposed an edge-enhanced hierarchical graph-to-tree model. Considering the heterogeneous issue and the long-distance dependencies of nodes in a graph representing a problem, Zhang et al.[33] proposed a novel hierarchical heterogeneous graph encoding method.

In recent years, pre-trained language models[34–36] have been applied in MWP solvers to obtain enriched text representations. Shen et al.[35] built a multitask framework to pre-train a language model that could address the limitation of minor mistakes in the generation of expression. Inspired by human-like hierarchical reasoning, Yu et al.[36] integrated the word-level and sentence-level reasoning by using outside knowledge from the pre-trained models.

Previous studies mainly constructed and utilized relationships between words, and some of them obtained richer semantic information with the support of external knowledge or pre-trained models. Comparatively, our work imitates the comprehension process of humans and we learn semantics and relationships between clauses from word-level to clause-level to improve the representation and understanding of the problem.

## 2.2 Clause-level dependency relationships

The clause-level relationship (CLR) is a basic concept in linguistics[37] and extensive research has been conducted in many studies[38–40]. Understanding the statement is a key phase of the math word problems solving process, and realizing the dependency relationships among clauses is the basis of understanding. The clause-level dependency relationship is the semantic dependency relationship between acts, events or states represented syntactically by syndetically connected finite clauses[39]. Dependency relationships can be judged by the importance of the connection between two clauses. Text comprehension is complex and supported by a variety of lower-level and higher-level processes that aim to understand the words, the sentences, and the relationships between the sentences[41].

Currently, studies are mainly carried out from the perspective of linguistics, such as the interpretability of clause relationships based on surface-syntactic information[39]. Some studies explore the challenges of solving mathematical word problems with the observation of the linguistic features of words, clauses and sentences[12]. However, there is much research about clause-level relationships in linguistics, but they are not generally applied in automatically solving MWP.

# 3 CLRSolver

## 3.1 Problem definition

In this subsection, we formally introduce the math word problem solving. Generally, a math word problem $(P, E_P, s_P)$ consists of a problem text $P$, a mathematical expression $E_P$ and an answer $s_P$.

1) The problem text $P$ is a sequence of $n$ word tokens and numeric values: $P = \{p_1, p_2, \cdots, p_n\}$, where $p_i$ is a word token (e.g., "dollars" in Fig. 1) or a numeric value (e.g., "80"). We denote the numeric values set for the problem $P$ as $N_P$ (e.g., $\{80, 0.8\%, 0.25\%\}$).

2) The mathematical expression $E_P = \{y_1, y_2, \cdots, y_m\}$, which is a sequence of $m$ symbols. Each symbol $y_i$ comes from the decoding target vocabulary which is composed of the operators set $V_O$ (e.g., $\{+, -, \times, \div\}$), numeric constants set $V_C$ (e.g., $\{1, 2, \pi, \cdots\}$) and $N_P$, where $V_P = V_O \cup V_C \cup N_P$.

3) The answer of $P$ is a numeric value $s_P$ (e.g., "16") derived from the expression $E_P$.

The goal of the MWP solver is to train a model that reads tokens from the input sequence $P$ and generates the output sequence of the expression $E_P = \{y_1, y_2, \cdots, y_m\}$, based on which calculates the answer $s_P$.

## 3.2 Model overview

Fig. 2 shows the overall architecture of clause-level relationship-aware solver (CLRSolver), which consists of two main components: 1) A multi-view encoder for learning clause-level dependency relationships and enhancing the representation of problems. 2) A tree-based decoder for generating mathematical expressions based on the word and clause representations. We mainly focus on the former for a better understanding of problems by mimicking human comprehension from a lower level to a higher level. In lower processes, the encoder learns the semantics of words and clauses by a gated recurrent unit (GRU) network. In higher processes, the encoder learns the dependency relationships between clauses from the local view and global view with three novel modules. In the following subsections, we will describe each part of the CLRSolver in detail.

## 3.3 Multi-view encoder

Following the two processes in human comprehension, in the lower-level processes, the encoder reads the problem sequence and produces its word and clause representations, which encode its semantics. In the higher-level processes, the encoder learns the dependency relationships among clauses from the local view and the global

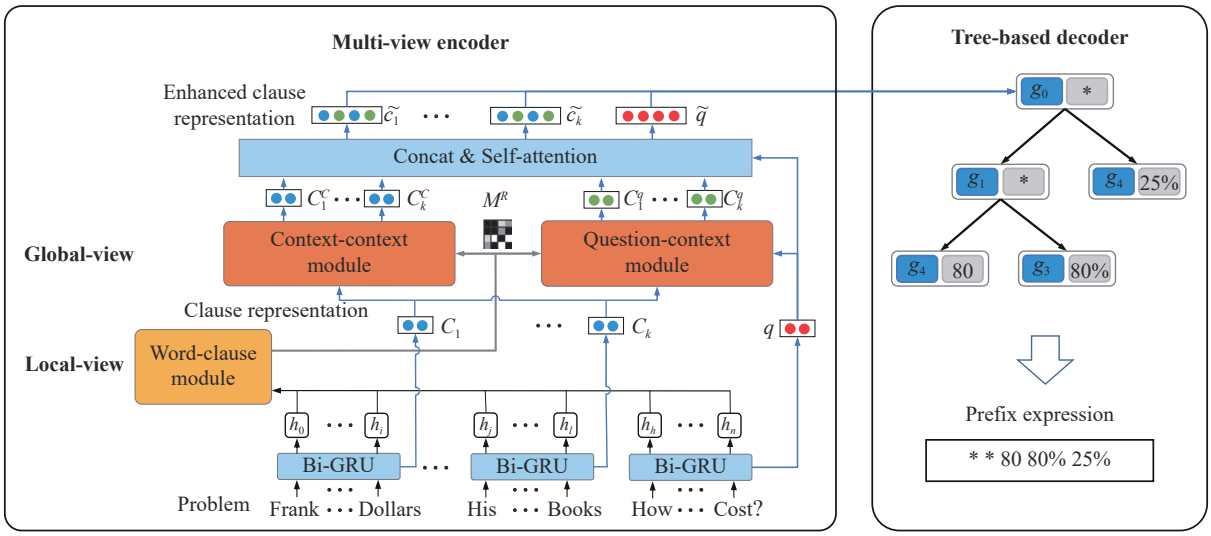



Fig. 2 The overview of CLRSolver framework

view respectively by three vital modules. In the local view, we propose a word-clause module for learning relationships according to the relevance of words in different clauses. In the global view, we propose a context-context module for enhancing context clause representations based on the learned relationships among context clauses and a question-context module for enhancing clause representations with the learned relationships between the question clause and context clauses.

### 3.3.1 Context-enriched word and clause embedding

In the lower-level processes, humans read text sequence and encode the representation of the words in the text and their semantics. We split the given problem $P = \{p_1, p_2, \cdots, p_n\}$ into $m$ clauses $C_P = \{C_1, \cdots, C_m, Q\}$ by commas and periods considering human reading habits and the consideration of semantic unit division in linguistics, where $C_i = \{p_{i1}, p_{i2}, \cdots, p_{il}\}$ is a subsequence of words from $P$ (shown in Fig. 1). The encoder first maps each word token $p_s$ to an embedding vector $\boldsymbol{x}_s$ pre-trained with word2vec[42]. We apply the word2vec method for a fair comparison with state-of-the-art (SOTA)'s scheme, and we realize that the pre-trained model bidirectional encoder representations from transformers (BERT)[43] can be applied for better performance. Then, the encoder learns the representation $\boldsymbol{h}_s$ and $\boldsymbol{c}_i$ for word $p_s$ and clause $C_i$ respectively with a bidirectional GRU. The context-enriched representation $\boldsymbol{h}_s$ of word token $p_s$ is generated as follows:

$$\boldsymbol{h}_s^f = \mathrm{GRU}_f(\boldsymbol{h}_{s-1}^f, \boldsymbol{x}_s) \tag{1}$$

$$\boldsymbol{h}_s^b = \mathrm{GRU}_b(\boldsymbol{h}_{s+1}^b, \boldsymbol{x}_s) \tag{2}$$

$$\boldsymbol{h}_s = \boldsymbol{h}_s^f + \boldsymbol{h}_s^b \ . \tag{3}$$

The context-enriched representation $\boldsymbol{c}_i$ of clause $C_i$ is generated as follows:

$$\boldsymbol{c}_i = \boldsymbol{h}_j + \boldsymbol{h}_k \tag{4}$$

where $\boldsymbol{h}_j$ and $\boldsymbol{h}_k$ are the representation vectors of the first token $p_{i1}$ and the last token $p_{il}$, which are generated by (3). Question representation $\boldsymbol{q}$ is generated the same as $\boldsymbol{c}_i$.

After obtaining $\boldsymbol{h}_s$, $\boldsymbol{c}_i$ and $\boldsymbol{q}$, inspired by the relationship establishment in higher-level processes of human local and global comprehension strategies, the encoder aims to update clause representation $\widetilde{\boldsymbol{c}}_i$ by understanding its relationships with other clauses. To this end, we propose modules to learn enhanced clause relationships in local view and global view.

### 3.3.2 Word-clause module

Humans discover the dependency relationship between two clauses by utilizing the connection of words in them. As illustrated in Fig. 1(b), objectively, if the "stationary" in clause $Q$ is changed to "books", the relationship between $Q$ and $C_3$ has changed absolutely, so the relevance of the word "stationary" in $Q$ and $C_3$ is the basis of the relationship. Moreover, if we exchange "stationery" with "books" with each other in clause $C_3$, the relationship features of $C_3$ in the problem would be changed. Therefore, the relevance of words in two clauses plays a non-negligible role in the relationships among clauses. To capture the dependency relationships between clauses from the local view (i.e., the word level), we first capture word-word connections and then aggregate these features with a CNN-based module inspired by [44] for its reliable performance in text matching and reservations on the position information. We apply cosine similarity to model the relevance $\boldsymbol{M}_{ij}^{kh}$ of two words in clause $C_k$ and $C_h$, and generate the relevance matrix $\boldsymbol{M}^{kh}$ (matrix in "Local-view" of Fig. 3), the element $\boldsymbol{M}_{ij}^{kh}$ of matrix $\boldsymbol{M}^{kh}$ is generated as follows:

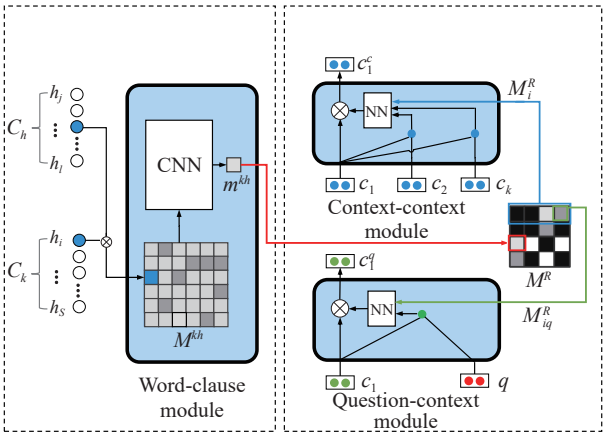



Fig. 3 Local-view and global-view architecture

$$\boldsymbol{M}_{ij}^{kh} = \text{cosine}(\boldsymbol{x}_i^k, \boldsymbol{x}_j^h). \tag{5}$$

As shown in Fig. 3, the input of CNN is the word relevance matrix $\boldsymbol{M}^{kh}$ of clause $C_k$ and $C_h$, the output is the relationship score which indicates the dependency relationship intensity between $C_k$ and $C_h$ in local view, $m^{kh}$ of is generated as follows:

$$m^{kh} = \text{CNN}(\boldsymbol{M}^{kh}). \tag{6}$$

In a problem, all the $m^{kh}$ for each clause pair compose the local-view dependency relationships matrix $\boldsymbol{M}^R$ for the problem, where the element $\boldsymbol{M}_{kh}^R = m^{kh}$.

After understanding the relationships in the local view, humans try to tease out the relations between conditions in the context clauses based on the overall understanding of clauses. Therefore, we need to capture the relationships in the global view from the perspective of their overall semantic representation $\boldsymbol{c}_i$. However, existing methods involving relationship exploration, such as attention mechanisms, cannot fit well in this scenario. In addition, since the context and question clauses take different roles in problem understanding, the relationships between context clauses and between the question and context clauses should be considered differently as mentioned before. Thus, we propose a context-context module for relationships between context clauses and a question-context module for relationships between question and context clauses. The goal of these two modules is to strengthen or weaken the original clause representation $\boldsymbol{c}_i$ according to the intensity of dependency relationships, which can explicate the significance of a clause in a problem.

### 3.3.3 Context-context module

The relationship between the context clauses is independent of the problem clauses. We intend to learn the relationships between them from the context clauses themselves, so we propose a context-context module individually. As illustrated in Fig. 3, the context-context module generates relationship weight $s_{ij}$ (blue dot in the context-context module) for clause $C_i$ with the rest of clauses $C_j$:

$$s_{ij}^c = \text{Softmax}(\boldsymbol{W}_{ca}[\boldsymbol{c}_i, \boldsymbol{c}_j] + b_{ca}) \tag{7}$$

where $\boldsymbol{W}_{ca}$ and $b_{ca}$ are learnable weight matrices and biases. Then, the module generates relationship weight $r_i^c$ with all the $s_{ij}$ from all context clauses and $\boldsymbol{M}_i^R$ from $\boldsymbol{M}^R$, then obtain the representation $\boldsymbol{c}_i^c$. $\boldsymbol{M}_i^R$ is the $i$th row elements $\boldsymbol{M}^R$ and indicates the local-view dependency relationships between clause $C_i$ and all the other clauses in a problem. The relationship weight $r_i^c$ indicate the importance of clause $C_i$ among all context clauses, and the generated $\boldsymbol{c}_i^c$ is the updated clause representation based on the overall relationships of context clauses, where

$$r_i^c = \text{Relu}(\boldsymbol{W}_{cb}[\boldsymbol{M}_i^R; s_{i1}^c; \cdots; s_{in}^c] + b_{cb}) \tag{8}$$

$$\boldsymbol{c}_i^c = r_i^c \boldsymbol{c}_i \tag{9}$$

where $\boldsymbol{W}_{cb}$ and $b_{cb}$ are learnable weight matrices and biases.

### 3.3.4 Question-context module

To solve the problem, humans figure out how the known contexts are associated with the question. Therefore, we introduce a question-context module to learn the dependency relationships between the question clause and context clauses from their overall understanding. Similar to the generation of $r_i^c$, we first learn the dependency relationship score $s_i^q$ between the question clauses $Q$ and the context clause $C_i$, then we combine the relationship scores from local and global-view to learn the relationship weight $r_i^q$. Next, to enhance the clause representation $\boldsymbol{c}_i$ to $\boldsymbol{c}_i^q$ for better comprehension, we generate $r_i^q$ with the relationship score $s_i^q$:

$$s_i^q = \text{Softmax}(\boldsymbol{W}_{qa}[\boldsymbol{c}_i, \boldsymbol{q}] + b_{qa}) \tag{10}$$

$$r_i^q = \text{Relu}(\boldsymbol{W}_{qb}[\boldsymbol{M}_{iq}^R; s_i^q] + b_{qb}) \tag{11}$$

where $\boldsymbol{W}_{qa}$, $\boldsymbol{W}_{qb}$, $b_{qa}$ and $b_{qb}$ are learnable weight matrices and biases.

New clause representation $\boldsymbol{c}_i^q$ is finally generated as follows:

$$\boldsymbol{c}_i^q = r_i^q \boldsymbol{c}_i. \tag{12}$$

After updating clause representation $\boldsymbol{c}_i$ from two views, considering a better understanding of clauses by fusing the representations from different views, we generate the concatenation of $\boldsymbol{c}_i^c$ and $\boldsymbol{c}_i^q$ as follows:

$$\boldsymbol{c}_r^* = \text{Concat}(\boldsymbol{c}_i^c; \boldsymbol{c}_i^q). \tag{13}$$

Finally, $\widetilde{\boldsymbol{c}_i}$ and $\widetilde{\boldsymbol{q}}$, the final representation of context clause $C_i$ and question clause $Q$, are generated by the attention module as follows:

$$S_{as}(\boldsymbol{c}_i^*, \boldsymbol{c}_k^*) = \boldsymbol{W}_{ss}^{\mathrm{T}} \text{Relu}(\boldsymbol{W}_{sa}^{\mathrm{T}}[\boldsymbol{c}_i^*, \boldsymbol{c}_k^*]) \tag{14}$$

$$w_k = \frac{\exp(S_{as}(\boldsymbol{c}_i^*, \boldsymbol{c}_k^*))}{\sum_j \exp((S_{as}(\boldsymbol{c}_i^*, \boldsymbol{c}_j^*))} \tag{15}$$

$$\boldsymbol{c}_i^r = \sum_k w_k \boldsymbol{c}_k^* \tag{16}$$

$$\widetilde{\boldsymbol{c}_l} = \text{Relu}(\boldsymbol{W}_{so}[\boldsymbol{c}_i^*; \boldsymbol{c}_i^r] + b_{so}) \tag{17}$$

where $\widetilde{\boldsymbol{c}_l}$, as shown in Fig. 2, is the representation enhanced by the clause-level dependency relationships feature.

## 3.4 Tree-based decoder

Our decoder, following the HMS[13] framework, generates expressions by adopting the pre-order traversal manner. In the process of expression tree construction, the centermost operator is produced first, and the decoder generates the left child node and repeats this process until the leaf node is generated. Subsequently, the decoder generates the right node recursively.

### 3.4.1 Goal decomposition and symbol prediction

In each symbol generation step, the decoder first generates a context vector $\boldsymbol{c}$ according to the goal vector $\boldsymbol{g}$. Then, the decoder predicts symbols by the pointer-generator network[45] according to the given goal and context vector. The decoding processes can be summarized as follows:

**Step 1. Root goal generation.** We initialize the goal vector with the question clause representation $\widetilde{\boldsymbol{q}}$. Given goal vector $\boldsymbol{g}_0 = \widetilde{\boldsymbol{q}}$, the decoder generates context representation that related to the goal vector to reduce the influence of irrelevant words and clauses and query relevant words and clauses. To explore the relevant parts of a problem and imitate the process of human solving problem from clause to word, we apply the hierarchy attention mechanism proposed in HMS. The context vector is generated as follows:

$$\boldsymbol{c} = \text{HATT}(\boldsymbol{g}_0, \widetilde{\boldsymbol{c}_1}, \boldsymbol{h}) \tag{18}$$

$$y_1 = \text{PointerGen}(\boldsymbol{g}_0, \widetilde{\boldsymbol{c}_1}, \boldsymbol{e}_y, \boldsymbol{h}_y) \tag{19}$$

where HATT is the hierarchical attention mechanism, PointerGen is the pointer generator network, $\boldsymbol{e}_y$ is a learnable embedding of the external symbol $y$ or an operator, $\boldsymbol{h}_y$ denotes the representation of quantity token $y$.

**Step 2. Left goal generation.** Then, the encoder generates the left goal $\boldsymbol{g}_l$ conditioned on the parent node $\boldsymbol{g}_p$ which predicted in last step:

$$\boldsymbol{g}_l = \text{Left}(y_p, \boldsymbol{c}_p, \boldsymbol{g}_p) \tag{20}$$

$$y_l = \text{PointerGen}(\boldsymbol{g}_l, \widetilde{\boldsymbol{c}_l}, \boldsymbol{e}_y, \boldsymbol{h}_y) \tag{21}$$

where $\boldsymbol{g}_p$, $y_p$, $\boldsymbol{c}_p$ stand for the goal vector, predicted token and context vector of parent node, respectively. After the left goal generation, the decoder predicts symbols $y_l$ for the current node.

**Step 3. Right goal generation.** Given the goal and context vectors in the left and parent nodes. First, the decoder generates the subtree embedding of its sibling node. Then, the decoder decomposes the goal and generates a new goal for the current node as

$$\boldsymbol{t}_l = \text{SubTree}(y_l, \boldsymbol{g}_l) \tag{22}$$

$$\boldsymbol{g}_r = \text{Right}(y_p, \boldsymbol{c}_p, \boldsymbol{g}_p, \boldsymbol{t}_l) \tag{23}$$

$$y_r = \text{PointerGen}(\boldsymbol{g}_r, \widetilde{\boldsymbol{c}_r}, \boldsymbol{e}_y, \boldsymbol{h}_y) \tag{24}$$

where $y_p$ is a predicted token in the expression and $\boldsymbol{t}_l$ is the tree embedding of the left goal. If token $y_r$ is a number, the algorithm backtracks to check whether the current goal could to be decomposed. If $y_r$ is an operator, the encoder will return to Step 2.

### 3.4.2 Model training

For each problem $P$ and corresponding target expression $E_P = \{y_1, y_2, \cdots, y_m\}$, the loss $L$ can be defined as minimizing the sum of the negative log likelihood of the probabilities of $m$ target symbols:

$$L = \sum_{t=1}^{m} -\log P_c(y_t \mid y_1, y_2, \cdots, y_{t-1}, P) \tag{25}$$

where $P_c$ is the probability distribution generated by pointer-generator network for each predicted symbol $y_i$.

# 4 Experiment

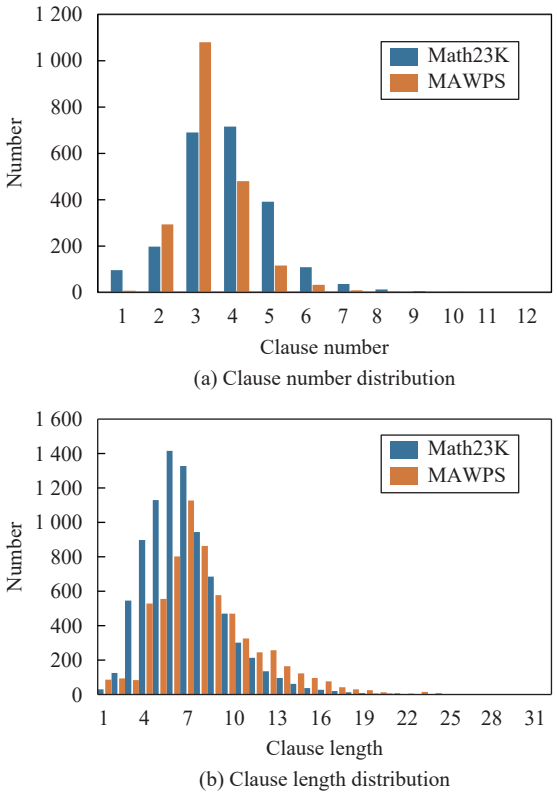
## 4.1 Datasets

We use two widely used datasets in our experiments: Math23K and MAWPS. Table 1 summarizes the statistics of the two datasets. Math23K is a benchmark dataset that contains 23 162 Chinese math word problems for elementary school students. MAWPS is an English dataset

Table 1 Datasets statistics

| Dataset | Num. problems | Num. vocabs | Avg. problem length | Avg.Num. clauses | Avg. expr. length |
|---|---|---|---|---|---|
| Math23K | 23 162 | 3 958 | 29 | 3.75 | 5.55 |
| MAWPS | 2 373 | 1 013 | 30 | 3.27 | 3.87 |

that contains problems with one or more unknown variables and contains 2 373 problems with one unknown variable in it. For a direct contrast, we randomly select 2 373 instances in Math23K (size of MAWPS) to summarize the statistics of the two datasets. In Fig. 4, we can find that the clause number of Math23K distributes more on the left side, which indicates the Math23K dataset is more difficult than MAWPS. Meanwhile, we find that the length of the clauses in MAWPS is longer, which is likely because the Chinese language is more concise in its presentation.




(a) Clause number distribution

(b) Clause length distribution

Fig. 4 Data analyses

## 4.2 Experimental setup

### 4.2.1 Implementation details

In CLRSolver, we set the dimension of word embedding and all hidden vectors as 128 and 512, respectively. The word embedding is initialized with pre-trained word2vec which is learned from the training dataset. Words that occur fewer than 5 times are converted to the token "UNK". Other parameters are initialized by Kaiming initialization[46]. We set the mini-batch as 64, dropout as 0.5 and learning rate as 0.001 to optimize the loss function (25). The learning rate is halved every 20 epochs. We apply the 5-fold cross-validation for MAWPS and follow the original setup of the partition in Math23K. All experiments are run on a Linux server with four 2.3 GHz Intel Xeon Gold 5 218 CPUs and a Tesla V100 GPU, and the PyTorch version is 1.8.1.

Since the performances are affected by the environments, for a fair comparison, we run all baselines in the same environment as described above with a 128-dimensional word embedding.

### 4.2.2 Baseline and evaluation

We compare our proposed model with several representative baseline models: Deep neural solver (NDS)[24] is a seq2seq model that directly translates the input problem into output expressions. T-RNN[28] applies a recursive neural network to predict the unknown operator in the predicted template. GROUP-ATT[8] applies a group attention mechanism to extract four features in input problems. Goal-driven tree-structured MWP solver (GTS)[5] generates expression by a tree-structured neural network to tree in a goal-driven manner. HMS[13] imitates human reading habits by exploiting the hierarchical word-clause-problem relation. Grap2Tree[9] applies a graph-based encoder to capture the relationships and order information among quantities. Edge-enhanced hierarchical graph-to-tree model (EEH-G2T)[30] constructs edge-labelled graphs for problem representation and captures common sense information based on external knowledge bases. Considering pre-trained word embedding in Baidu Encyclopedia in EEH-G2T, we changed it into pre-trained word2vec in the two datasets. Some methods[34–36] using the pre-training model have shown better results, we have not included them in the baseline for a fairer comparison since their performances are mainly determined by the corpus for pre-training. We rerun EEH-G2T without word embedding in Baidu Encyclopedia and retain its external knowledge bases.

To highlight the effectiveness of our relation-aware mechanisms in this domain and considering the widespread application of the attention mechanism for relationship extraction, such as image segmentation[47], and fault classification[48], and its convincing performance, we use attention to replace relationship-aware mechanisms in CLRSolver-ATT, as a variant of CLRSolver, for a contrast.

## 4.3 Experimental results

### 4.3.1 Answer performance

Table 2 reports the accuracy of all methods generated

Table 2 Answer accuracy of CLRSolver and baselines

| Dataset | Math23K[1] | MAWPS | Math23K*[2] |
|---|---|---|---|
| DNS | 0.584 | 0.598 | – |
| T-RNN | 0.665 | 0.670 | – |
| GROUP-ATT | 0.698 | 0.757 | – |
| GTS | 0.750 | 0.789 | 0.741 |
| Graph2Tree | 0.764 | 0.812 | 0.750 |
| HMS | 0.756 | 0.796 | 0.743 |
| EEH-G2T | 0.766 | **0.833** | 0.754 |
| CLRSolver-ATT | 0.751 | 0.790 | 0.742 |
| **CLRSolver** | **0.769** | 0.822 | **0.758** |
| CLRSolver w/o L | 0.763 | 0.807 | – |
| CLRSolver w/o G | 0.759 | 0.793 | – |

[1]Math23K denotes the results on the public test set.
[2]Math23K* denotes 5-fold cross-validation.

in the environment in Section 4.2.1 of different models, and there are several observations. First, our model outperforms all baseline models except EEH-G2T, which demonstrates that CLRSolver can effectively understand the problem by exploiting dependency relationships between clauses, and thus enhance expression inference. Second, EEH-G2T performs best in the MAWPS dataset and the accuracy of the other datasets is close to that of our model, which we believe is the contribution of external knowledge bases. Third, CLRSolver performs better than HMS, for the same decoder structure, which shows that the importance of clause-level dependency relationships exploitation and application. Last, the performance of tree-based methods (GTS, HMS) or graph-based methods(Grap2Tree, EEH-G2T) are far ahead of these models(DNS, T-RNN, GROUP-ATT) evolved from the seq2seq model. This is probably because more relationship features, such as word and clause relationships, can enhance the understanding of problems to models.

We compare the results of our model and Graph2Tree on 5-fold cross-validation on the Math23K dataset (Math23K* in Table 2). The results show that the superiority of performance expends in 5-fold cross-validation, which demonstrates the generalization of our model is better.

We also conduct an independent samples t-test to compare the results of our model and Graph2Tree on 5-fold cross-validation on the Math23K dataset. The $p$-value is 0.004 9 ($< 0.05$), which shows that our improvement is statistically significant.

#### 4.3.2 Ablation study

To verify the effects of various components in our model, the context-context module, the question-context module and the word-clause module, we conduct ablation studies. The results are shown in Table 2. Specifically, "CLRSolver w/o L" directly implements two global-view modules without the word-clause module. "CLRSolver w/o G" applies the word-clause module without two global view modules. Obviously, the accuracy of our solver degrades when any module is missing, which means that all modules are contribute and complete in the problem solving.

#### 4.3.3 Performance over clause number and problem length

Considering the possible impact of clause number on the performance in modelling dependency relationships, we analyse the increasing number of clauses in the test set, and the results are illustrated in Fig. 5. The accuracies of the test instances are compared with the Graph2Tree model. Since the results on the two datasets are approximate, we only report the results on Math23K for simplification. There are several observations. First, with the increase in clause number, the performances of all models follow an accuracy descending pattern, because of the increase in problem difficulty. Second, we note that our model outperforms Grap2Tree when the clause number is greater than 2. We realize that the more clauses there are in a problem, the relationships between clauses are more important and necessary to clearly comprehend the problem.

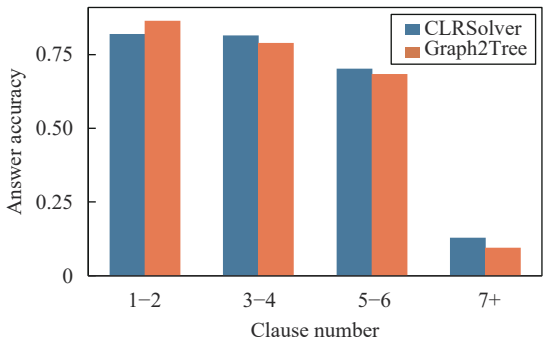



Fig. 5 Accuracy over clause number

The reason may be that the more clauses there are, the more difficult the problem is. In complex problems, the relationship between clauses is more important to solving the problem. When there are fewer clauses, there is less dependence on the relationship between clauses. In addition, for a longer problem, there could be more clauses that do not participate in the solving, and there could be some clauses that are background information clauses. Our mechanism weakens the relationship between such clauses and other clauses, so a better performance can be achieved.

#### 4.3.4 Performance over the number of operators and problem length

To investigate the performance of models with the increasing length of expressions and problem text, we report the number of test instances over different operator numbers in Fig. 6 and different problem lengths in Fig. 7 with Graph2Tree.

In Fig. 6, the horizontal axis is the number of operat-

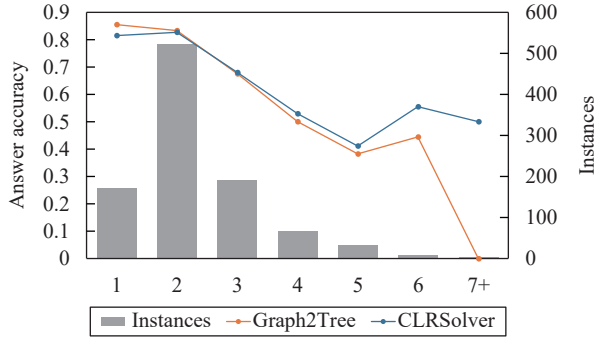



Fig. 6 Accuracy for the increasing number of operators in expressions

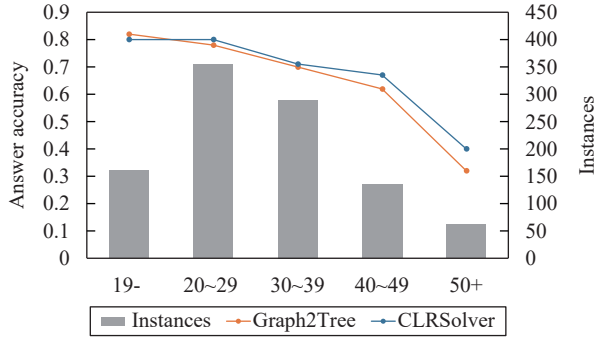



Fig. 7 Accuracy over problem length

ors that indicates the length of the expression, the left vertical axis is the corresponding answer accuracy, and the right vertical axis is the corresponding number of expressions. We can obtain the following observations: 1) Similar to the accuracy performance with clause number in Fig. 5, the answer accuracy represents a downward trend with the increasing number of operators (length of expressions). 2) CLRSolver has no notable superiority when the number of operators is less than 3, but our model has a better performance and continues to expand it with the increase of operators. As an operator is related to two quantities that occur in two clauses, the result demonstrates the advantage of our model in solving complex and difficult problems where the relationships among clauses are complicated.

Similar trends are observed in Fig. 7. The accuracy performance of the two models decreases and the difference between the two models increases with the increasing problem length, which corresponds to increasing difficulty. This shows that our model is superior in handling more complex problems as mentioned in Section 4.3.3.

#### 4.3.5 Case study

Finally, we conduct case studies on the expression generated by Graph2Tree and our model, and we visualize the dependency relationships our model learned from the perspectives of local-view to global-view respectively in Fig. 8. We visualize the relationship weight of the attention mechanism and our method in Fig. 9 to provide a better understanding of the difference between the attention mechanism and our method.

Selected cases are shown in Fig. 8. We divide the score of relationships into three levels: strong, moderate, and weak, and the corresponding values are within 0.5, 0.5 to 0.75, and above 0.75. The matrix $\boldsymbol{L}_C$ and $\boldsymbol{L}_Q$ are a part of the local view relationship matrix $\boldsymbol{M}^R$, which means the context-context and query-context relationships in the local view. The element of matrix $\boldsymbol{G}_C$ and $\boldsymbol{R}_C$ is generated in (7) and (8), respectively. $\boldsymbol{G}_C$ and $\boldsymbol{G}_Q$ are the global-view relationship weight matrices generated from (7) and (10), respectively. $\boldsymbol{R}_C$ and $\boldsymbol{R}_Q$ are the final relationship intensity matrices, generated from (8) and (11), respectively, which indicate the relationship strength from the local and global views. Obviously, we can observe that the relationships in relationship matrices between clauses learned by local-view and global-view are different, which demonstrates that our model can learn the dependency relationships between two clauses in different views.

In Problem 1, "each" cost is associated with the "books" in clause $C_2$. Our model predicts the operator "$\times$" right which indicates that our model can learn the relationships between this strong connection based on the overall semantics of the two clauses, which is demonstrated in the global relationship matrix $\boldsymbol{G}_C$.

In Problem 2, to understanding this problem, MWP solvers should know that the "dictionary" and "cookbook" are the books asked in the question and realize the connection between clause $\boldsymbol{C}_3$ and other clauses. The local-view relationship matrix $\boldsymbol{M}^R$ and the global-view relationship matrix $\boldsymbol{R}_C$ both show the strong connection between clauses $\boldsymbol{C}_3$ and other clauses. Grap2Tree missed "$n_3$" because it could not build the connection.

In the last case in Fig. 8, the global-view relationship matrix $\boldsymbol{R}_C$ indicates that our model can learn the relationship between $C_2$ and $C_3$ from the overall semantics. We can find the strong connections our model learned between question clause and $C_2$, $C_3$ in matrix $\boldsymbol{R}_Q$. In addtion, our model also learned that the relationships between $C_4$ and other clauses are weak because this clause is background information. In contrast, Grap2Tree gives a wrong operation prediction "–" as Grap2Tree cannot realize the relationships among question $Q$ ("How $\cdots$ altogether") and context clauses $C_2$ ("$n_1$ $\cdots$ left over") $C_3$ ("and$\cdots$left over").

The attention weight matrices and relationship weight matrices in Fig. 9 shows the different effects in relationship feature learning. In Problem 4, we can easily realize that clause $C_1$ is background information that is not decisive to the problem-solving. Comparatively, clause $C_1$ still has a high relative weight with clause $Q$ in attention. Our model woke the connection between $C_1$ and other clauses and enhanced the connection between context clauses. In Problem 5, clause $C_1$ is useless background information, and clause $C_1$ is not truly necessary for problem-solving. Apparently, our mechanism reduced the rela-

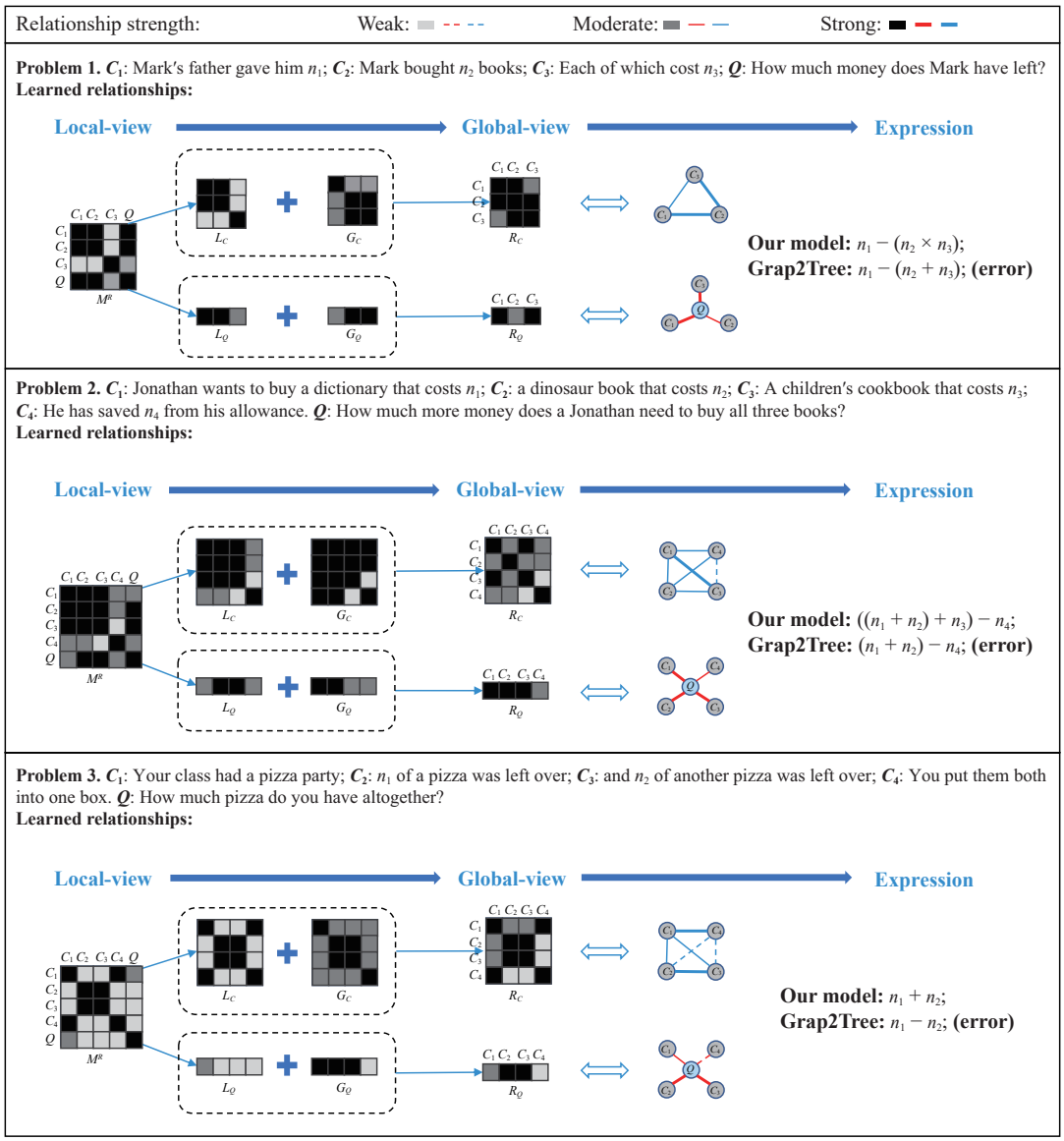



Fig. 8 Three examples of MWP solving with our CLRSolver model

tionship strength of both $C_1$ and $C_5$, and enhanced the intensity of the relationship of more important clauses such as $C_3$ and $C_4$. Comparatively, although the attention mechanism can also capture the different strengths of clause relationships, the strength of the relationship is not so reasonable in math word problem-solving. It reveals the effectiveness of simulating the human comprehension process in our mechanism when compared with the attention mechanism.

## 5 Discussions

From the above experiments, it is clear that mimicking the process of human comprehension from a lower level to a higher level would enhance the comprehension of a problem. First, splitting problems into clauses not only conforms to human's intuition of reading but also enables better comprehension based on clause-level relationships. Second, learning the relationship between clauses from a local view to a global view is also significantly beneficial for understanding and solving problems. The impressive performance of CLRSolver can be seen among all two widely used datasets, which highlights the importance of clause relationships and the superiority of CLRSolver.

There are still some directions for future studies. First, better word representations and external knowledge could contribute to the reasoning of the model, so we could apply pre-trained models or introduce external knowledge in the future explorations. Second, we are imitating the comprehension process of human reading. In the following work, we could combine the learned relationship and apply it to the reasoning process in the decoder. Third, there are other relationships in math word problems, such as the relationships between the numbers and entities, combining these relationships with clause-

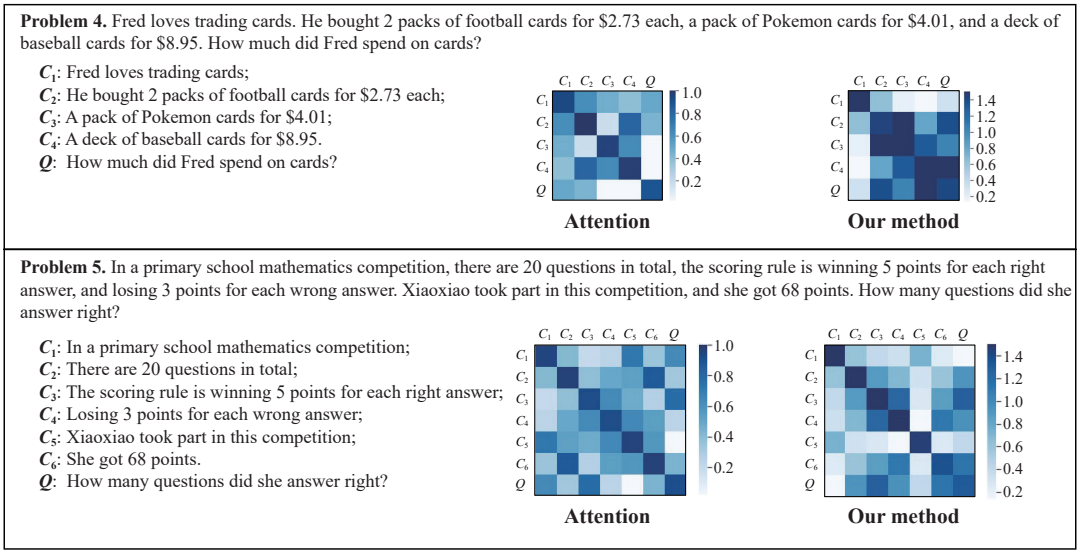



Fig. 9 Examples of relationship weights in the attention mechanism and our CLRSolver model

level relationships would be a creditable attempt and a challenging task in the future.

## 6 Conclusions

In this paper, we propose a novel clause-level relationship-aware solver (CLRSolver) to solve math word problems by mimicking the lower-lewer and higher-level processes of human comprehension. Specifically, in the lower-level process, we learn the semantics of words and clauses. In the higher-level process, inspired by the local and global view of humans in relationship construction, we first developed a CNN-based module for capturing the dependency relationships among clauses from a local view. Then, we designed two modules to learn the context-context and question-context dependency relationships from a global view. After that, the model generates enhanced clause representations, based on the learned relationships, to predict solution expressions. Experimental results demonstrated the effectiveness of CLRSolver.

## Acknowledgements

This work was supported by National Key Research and Development Program of China (No. 2021YFF090 1003), and National Natural Science Foundation of China (Nos. 61922073, U20A20229, and 62106244). The authors wish to thank the anonymous reviewers for their helpful comments.

## References

[1] T. Brants. Natural language processing in information retrieval. In *Proceedings of Computational Linguistics in the Netherlands*, University of Antwerp, Antwerp, Belgium, pp. 1–13, 2003.

[2] Y. K. Xian, Z. H. Fu, S. Muthukrishnan, G. De Melo, Y. F. Zhang. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, Paris, France, pp. 285–294, 2019. DOI: 10.1145/3331184.3331203.

[3] D. X. Zhang, L. Wang, L. M. Zhang, B. T. Dai, H. T. Shen. The gap of semantic parsing: A survey on automatic math word problem solvers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 9, pp. 2287–2305, 2019. DOI: 10.1109/TPAMI.2019.2914054.

[4] D. P. Huang, S. M. Shi, C. Y. Lin, J. Yin. Learning fine-grained expressions to solve math word problems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Copenhagen, Denmark, pp. 805–814, 2017. DOI: 10.18653/v1/D17-1084.

[5] Z. P. Xie, S. C. Sun. A goal-driven tree-structured neural model for math word problems. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence Main track*, Macao, China, pp. 5299–5305, 2019. DOI: 10.24963/ijcai.2019/736.

[6] Y. N. Hong, Q. Li, D. Ciao, S. Y. Huang, S. C. Zhu. Learning by fixing: Solving math word problems with weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 6, pp. 4959–4967, 2021.

[7] S. Roy, D. Roth. Unit dependency graph and its application to arithmetic word problem solving. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, ACM, San Francisco, USA, pp. 3082–3088, 2017.

[8] J. R. Li, L. Wang, J. P. Zhang, Y. Wang, B. T. Dai, D. X. 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*, Florence, Italy, pp. 6162–6167, 2019. DOI: 10.18653/v1/P19-1619.

[9] J. P. Zhang, L. Wang, R. K. W. Lee, Y. Bin, Y. Wang, J. Shao, 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*, ACL, pp. 3928–3937, 2020. DOI: 10.18653/v1/2020.acl-main.362.

[10] D. A. Balota, G. B. F. d'Arcais, K. Rayner. *Comprehension Processes in Reading*. New York, USA: Routledge, 1990. DOI: 10.4324/9780203052389.

[11] T. A. Van Dijk, W. Kintsch. *Strategies of Discourse Comprehension*. New York, USA: Academic Press, 1983.

[12] M. Adoniou, Q. Yi. Language, mathematics and English language learners. *The Australian Mathematics Teacher*, vol. 70, no. 3, pp. 3–13, 2014.

[13] X. Lin, Z. Y. Huang, H. K. Zhao, E. H. Chen, Q. Liu, H. Wang, S. Wang. HMS: A hierarchical solver with dependency-enhanced understanding for math word problem. In *Proceedings of AAAI Conference on Artificial Intelligence*, vol. 35, no. 5, pp. 4232–4240, 2021.

[14] E. A. Feigenbaum, J. Feldman. *Computers and Thought*. New York, USA: McGraw-Hill, 1963.

[15] D. G. Bobrow. Natural Language Input for a Computer Problem Solving System, Series/Report no. AITR-219, Department of Mathematics, Massachusetts Institute of Technology, Cambridge, USA, 1964.

[16] J. R. Slagle. Experiments with a deductive question-answering program. *Communications of the ACM*, vol. 8, no. 12, pp. 792–798, 1965. DOI: 10.1145/365691.365960.

[17] C. R. Fletcher. Understanding and solving arithmetic word problems: A computer simulation. *Behavior Research Methods, Instruments, & Computers*, vol. 17, no. 5, pp. 565–571, 1985. DOI: 10.3758/bf03207654.

[18] Y. Bakman. Robust understanding of word problems with extraneous information. [Online], Available: https://arxiv.org/pdf/math/0701393.pdf, 2007.

[19] N. Kushman, Y. Artzi, L. Zettlemoyer, R. Barzilay. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, pp. 271–281, 2014. DOI: 10.3115/v1/P14-1026.

[20] A. Mitra, C. Baral. Learning to use formulas to solve simple arithmetic problems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 2144–2153, 2016. DOI: 10.18653/v1/P16-1202.

[21] R. Koncel-Kedziorski, H. Hajishirzi, A. Sabharwal, O. Etzioni, S. D. Ang. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 585–597, 2015. DOI: 10.1162/tacl_a_00160.

[22] S. M. Shi, Y. H. Wang, C. Y. Lin, X. J. Liu, Y. Rui. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, pp. 1132–1142, 2015. DOI: 10.18653/v1/D15-1135.

[23] D. Q. Huang, S. M. Shi, C. Y. Lin, J. Yin, W. Y. Ma. How well do computers solve math word problems? Large-scale dataset construction and evaluation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 887–896, 2016. DOI: 10.18653/v1/P16-1084.

[24] Y. Wang, X. J. Liu, S. M. Shi. Deep neural solver for math word problems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 845–854, 2017. DOI: 10.18653/v1/D17-1088.

[25] L. Wang, D. X. Zhang, L. L. Gao, J. K. Song, L. Guo, H. T. Shen. MathDQN: Solving arithmetic word problems via deep reinforcement learning. In *Proceedings of AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, pp. 5545–5552, 2018. DOI: 10.1609/aaai.v32i1.11981. DOI: 10.1609/aaai.v32i1.11981.

[26] L. Wang, Y. Wang, D. Cai, D. X. Zhang, X. J. Liu. Translating a math word problem to an expression tree. [Online], Available: https://arxiv.org/pdf/1811.05632.pdf, 2018.

[27] T. R. Chiang, Y. N. Chen. Semantically-aligned equation generation for solving and reasoning math word problems. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Minneapolis, Minnesota, pp. 2656–2668, 2018. DOI: 10.18653/v1/N19-1272.

[28] L. Wang, D. X. Zhang, J. P. Zhang, X. Xu, L. L. Gao, B. T. Dai, H. T. Shen. Template-based math word problem solvers with recursive neural networks. In *Proceedings of AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 7144–7151, 2019. DOI: 10.1609/aaai.v33i01.33017144. DOI: 10.1609/aaai.v33i01.33017144.

[29] Q. Z. Wu, Q. Zhang, J. L. Fu, X. J. Huang. A knowledge-aware sequence-to-tree network for math word problem solving. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 7137–7146, 2020. DOI: 10.18653/v1/2020.emnlp-main.579.

[30] Q. Z. Wu, Q. Zhang, Z. Y. Wei. An edge-enhanced hierarchical graph-to-tree network for math word problem solving. In *Proceedings of the Findings of the Association for Computational Linguistics*, Punta Cana, Dominican Republic, pp. 1473–1482, 2021. DOI: 10.18653/v1/2021.findings-emnlp.127.

[31] M. Yuhui, Z. Ying, C. Guangzuo, R. Yun, H. Ronghuai. Frame-based calculus of solving arithmetic multi-step addition and subtraction word problems. In *Proceedings of the Second International Workshop on Education Technology and Computer Science*, IEEE, Wuhan, China, pp. 476–479, 2010. DOI: 10.1109/ETCS.2010.316.

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

[33] Y. Zhang, G. Y. Zhou, Z. W. Xie, 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. DOI: 10.1109/TASLP.2022.3145314.

[34] Z. W. Liang, J. P. Zhang, L. Wang, W. Qin, Y. S. Lan, J. Shao, X. L. Zhang. MWP-BERT: Numeracy-augmented pre-training for math word problem solving. Available: https://aclanthology.org/2022.findings-naacl.74.pdf, 2022.

[35] J. H. Shen, Y. C. Yin, L. Li, L. F. Shang, X. Jiang, M. Zhang, Q. Liu. Generate & Rank: A multi-task framework for math word problems. In *Proceedings of the Findings of*

*the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic, pp. 2269–2279, 2021. DOI: 10.18653/v1/2021.findings-emnlp.195.

[36] W. J. Yu, Y. P. Wen, F. D. Zheng, N. Xiao. Improving math word problems with pre-trained knowledge and hierarchical reasoning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Punta Cana, Dominican Republic, pp. 3384–3394, 2021. DOI: 10.18653/v1/2021.emnlp-main.272.

[37] M. A. K. Halliday, C. M. I. Matthiessen. *An Introduction to Functional Grammar*. London, UK: Routledge, 2014.

[38] J. Ng, K. Lee, K. H. Khng. Irrelevant information in math problems need not be inhibited: Students might just need to spot them. *Learning and Individual Differences*, vol. 60, pp. 46–55, 2017. DOI: 10.1016/j.lindif.2017.09.008.

[39] K. Barker, S. Szpakowicz. Interactive semantic analysis of clause-level relationships. In *Proceedings of the 2nd Conference of the Pacific Association for Computational Linguistics*, Brisbane, Australia, pp. 22–30, 1995.

[40] T. Ohno, S. Matsubara, H. Kashioka, T. Maruyama, H. Tanaka, Y. Inagaki. Dependency parsing of Japanese monologue using clause boundaries. *Language Resources and Evaluation*, vol. 40, no. 3, pp. 263–279, 2006. DOI: 10.1007/s10579-007-9023-y.

[41] D. S. McNamara, J. Magliano. Toward a comprehensive model of comprehension. *Psychology of Learning and Motivation*, vol. 51, pp. 297–384, 2009. DOI: 10.1016/S0079-7421(09)51009-2.

[42] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, ACM, Lake Tahoe, Nevada, pp. 3111–3119, 2013.

[43] J. Devlin, M. W. Chang, K. Lee, K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Minneapolis, Minnesota, pp. 4171–4186, 2018. DOI: 10.18653/v1/N19-1423.

[44] L. Pang, Y. Y. Lan, J. F. Guo, J. Xu, S. X. Wan, X. Q. Cheng. Text matching as image recognition. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, ACM, Phoenix, USA, pp. 2793–2799, 2016.

[45] O. Vinyals, M. Fortunato, N. Jaitly. Pointer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, ACM, Montreal, Canada, pp. 2692–2700, 2015.

[46] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, Santiago, Chile, pp. 1026–1034, 2015. DOI: 10.1109/ICCV.2015.123.

[47] C. Z. Wu, J. Sun, J. Wang, L. F. Xu, S. Zhan. Encoding-decoding network with pyramid self-attention module for retinal vessel segmentation. *International Journal of Automation and Computing*, vol. 18, no. 6, pp. 973–980, 2021. DOI: 10.1007/s11633-020-1277-0.

[48] L. J. Zhou, J. W. Dang, Z. H. Zhang. Fault classification for on-board equipment of high-speed railway based on attention capsule network. *International Journal of Automation and Computing*, vol. 18, no. 5, pp. 814–825, 2021. DOI: 10.1007/s11633-021-1291-2.

**Chang-Yang Wu** received the B. A. degree in sport English from Shanghai University of Sport, China in 2014. Now, he is a master student in computer science at School of Data Science, University of Science and Technology of China (USTC), China.

His research interests include data mining and intelligent education systems.

E-mail: justwu@mail.ustc.edu.cn
ORCID iD: 0000-0002-7735-1568

**Xin Lin** received the B. Eng. degree in computer science from University of Science and Technology of China, China in 2019. He is currently a Ph. D. degree candidate in computer science at School of Computer Science and Technology at USTC, China. He has published papers in referred conference proceedings, such as AAAI 2021.

His research interests include data mining, math word problems, intelligent education systems.

E-mail: linx@mail.ustc.edu.cn

**Zhen-Ya Huang** received the B. Eng. degree in software engineering from Shandong University, China in 2014 and the Ph. D. degree in applied computer technology from University of Science and Technology of China, China in 2020. He is currently an associate researcher of School of Computer Science and Technology, USTC, China. He has published more than 30 papers in refereed journals and conference proceedings including TKDE, TOIS, KDD, AAAI. He has served regularly in the program committees of a number of conferences, and is reviewer for the leading academic journals.

His research interests include data mining, knowledge discovery, representation learning and intelligent tutoring systems.

E-mail: huangzhy@ustc.edu.cn

**Yu Yin** received the B. Sc. degree in computer science and technology from School of Computer Science and Technology, USTC, China in 2017. He is currently a Ph. D. degree candidate in computer science at School of Computer Science and Technology at USTC, China. He won the first prize in the Second Student Remote Direct Memory Access Programming Competition, in 2014. He has published papers in journals and conference proceedings related with data mining and machine learning, such as AAAI, KDD, ICDM, CIKM, SIGIR and ACM TKDE.

His research interests include data mining, intelligent education systems and reinforcement learning.

E-mail: yxonic@mail.ustc.edu.cn

**Jia-Yu Liu** received the B. Sc. degree in applied mathematics from USTC, China in 2020. Now, he is a master student in data science (computer science and technology) at School of Data Science, University of Science and Technology of China.

His research interests include data mining and intelligent education systems.

E-mail: jy251198@mail.ustc.edu.cn

**Qi Liu** received the Ph. D. degree in computer science from USTC, China in 2013. He is currently a professor at USTC, China. His general area of research is data mining and knowledge discovery. He has published prolifically in refereed journals and conference proceedings, e.g., the *IEEE Transactions on Knowledge and Data Engineering*, the *ACM Transactions on Information Systems*, the *ACM Transactions on Knowledge Discovery from Data*, the *ACM Transactions on Intelligent Systems and Technology*, KDD, IJCAI, AAAI, ICDM, SDM, and CIKM. He has served regularly in the program committees of a number of conferences, and is a reviewer for the leading academic journals in his fields. He is a member of the ACM, the IEEE, and the Alibaba DAMO Academy Young Fellow. His research is also supported by the National Science Fund for Excellent Young Scholars and the Youth Innovation Promotion Association of Chinese Academy of Sciences (CAS).

His research interests include data science, data mining, machine learning: methods and applications, recommender systems, social network analysis.

E-mail: qiliuql@ustc.edu.cn (Corresponding author)

ORCID iD: 0000-0001-6956-5550

**Gang Zhou** received the B. Eng. and M. A.Eng degrees in computer software from Information Engineering University, China in 1996 and 1999, the Ph. D. degree in computer software and theory from Beihang University, China in 2007. He was with the State key of Laboratory of Mathematical Engineering And Advanced Computing, Information Engineering University, as a research fellow.

His research interests are big data, knowledge graph, and data mining.

E-mail: gzhougzhou@126.com

ORCID iD: 0000-0001-8609-3954