Published in ACM Transactions on Knowledge Discovery from Data (2024) 18. DOI: 10.1145/3639569

# Math Word Problem Generation via Disentangled Memory Retrieval

WEI QIN, XIAOWEI WANG, and ZHENZHEN HU, Key Laboratory of Knowledge Engineering with Big Data(Hefei University of Technology), Ministry of Education, Hefei, China and Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China
LEI WANG, The School of Computing and Information Systems, Singapore Management University, Singapore, Singapore
YUNSHI LAN, The school of Data Science and Engineering, East China Normal University, Shanghai, China
RICHANG HONG, Key Laboratory of Knowledge Engineering with Big Data(Hefei University of Technology), Ministry of Education, Hefei, China and Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China

The task of math word problem (MWP) generation, which generates an MWP given an equation and relevant topic words, has increasingly attracted researchers' attention. In this work, we introduce a simple memory retrieval module to search related training MWPs, which are used to augment the generation. To retrieve more relevant training data, we also propose a disentangled memory retrieval module based on the simple memory retrieval module. To this end, we first disentangle the training MWPs into logical description and scenario description and then record them in respective memory modules. Later, we use the given equation and topic words as queries to retrieve relevant logical descriptions and scenario descriptions from the corresponding memory modules, respectively. The retrieved results are then used to complement the process of the MWP generation. Extensive experiments and ablation studies verify the superior performance of our method and the effectiveness of each proposed module. The code is available at https://github.com/mwp-g/MWPG-DMR.

CCS Concepts: • **Computing methodologies → Natural language generation;**

Additional Key Words and Phrases: Memory, retrieval, math word problem, text generation.

**ACM Reference Format:**
Wei Qin, Xiaowei Wang, Zhenzhen Hu, Lei Wang, Yunshi Lan, and Richang Hong. 2024. Math Word Problem Generation via Disentangled Memory Retrieval. *ACM Trans. Knowl. Discov. Data.* 18, 5, Article 123 (March 2024), 21 pages. https://doi.org/10.1145/3639569

Authors' addresses: W. Qin, Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Tuxin Road No. 193, Hefei, China, 230009; e-mail: qinwei.hfut@gmail.com; X. Wang, Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Tuxin Road No. 193, Hefei, China, 230009; e-mail: wxw.hfut@gmail.com; Z. Hu, Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Tuxin Road No. 193, Hefei, China, 230009; e-mail: huzhen.ice@gmail.com; L. Wang, The School of Computing and Information Systems, Singapore Management University, Singapore, Singapore, 178902; e-mail: lei.wang.2019@phdcs.smu.edu.sg; Y. Lan, The School of Data Science and Engineering, East China Normal University, Shanghai, China, 200241; e-mail: yslan@dase.ecnu.edu.cn; R. Hong (Corresponding author), Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Tuxin Road No. 193, Hefei, China, 230009; e-mail: hongrc.hfut@gmail.com.

## 1 INTRODUCTION

The **Math Word Problem (MWP)** plays an important role in mathematics education, since they are broadly used to assess and improve students' understanding of mathematical concepts and skills of solving math problems [52–54, 59, 67]. As shown in Table 1, an MWP consists of a question and a corresponding equation, and the question is composed of the *scenario description* marked by the orange color and the *logical description* marked by the cyan color. Students could strengthen their problem solving skills by learning from questions with the same logical description but different scenario description [52]. Many studies [16, 17, 48] have showed that high-quality MWPs could improve the teaching outcomes. However, manually designing MWPs by experts costs a lot, and the qualities of the generated MWPs heavily rely on the experts.

In this paper, we focus on the task of **Math Word Problem Generation (MWPG)**, which is to generate a MWP conditioned on both topic words and an equation. Traditional methods usually heuristically generate MWPs, based on some pre-defined text templates [7, 41, 45, 61]. However, the diversity of MWPs generated by text templates are not high as expected. Recently, some methods based on deep neural networks have brought significant improvement in generating MWPs. MaGNET [69], based on a standard encoder-decoder architecture, forces the entities in the generated MWP to correspond to the variables in inputs. The works in References [36] fuses information from equations and commonsense knowledge to facilitate the generation. And a recent work [59], based on a large-scale pre-trained language model, introduces an equation consistency constraint, which encourages the generated MWP to contain the exact same equation as the one used to generate it. However, the generation modules of those methods are only conditioned on the limited input topic words and equation, which might lead to that the scenario description of the generated MWP often lacks richness and the logical description of the generated MWP usually does not match the input equation. As shown in Figure 1(a), the generation of *seq2seq* lacks some keywords (such as *farm* for scenario description and *more than* for logical description).

To solve this problem, we first introduce a **simple memory retrieval (SMR)** module, which takes full advantage of the training MWPs, into the MWPG framework. The memory retrieval module has been demonstrated to enhance a variety of text generation tasks, such as open-domain question answering [4, 18, 23], dialogue response generation [1, 2, 27, 60, 63], and machine translation [3]. In specific, we record all the training MWPs into a **simple memory (SM)** in advance. During inference, we utilize the joint query (i.e., both topic words and the equation) to retrieve the **simple memory (SM)**. The retrieved MWPs are used to augment the generation condition. Since the logical description and the scenario description are entangled and form the MWP, the retrieved MWPs with matching logical description may contain mismatched scenario description or vice versa. As shown in Figure 1(b), retrieved by the joint query, the first retrieved result introduces a new keyword *farm* corresponding to *ducks* and *chickens*, improving the richness of the generated MWP's scenario description. However, it also introduces a new logical description keyword *less* to augment the generation condition. This induces the mismatch of the generated MWP and the input equation without any subtract operation. Similarly, the second retrieved result introduces *times* corresponding to the multiplication sign in the equation but also introduces *library*, which does not match with the scenario description(*ducks* and *chickens*).

To avoid introducing extra mismatched information in the retrieved results, based on the SMR, we further propose a **disentangled memory retrieval (DMR)** framework. Instead of directly building the SM, we first disentangle the training MWPs into the scenario description (orange part in Table 1) and the logical description (cyan part in Table 1). The scenario description is composed of several topic words. The logical description describes the information of the equation. Then, we record the scenario description and the logical description to build the **scenario description memory (SDM)** and the **logical description memory (LDM)**, respectively. During

Table 1. An Example of MWP

| | |
|---|---|
| MWP: | There are $N_0$ ducks in the farm, and chickens are $N_2$ more than $N_1$ times of ducks.How many chickens and ducks are there in total? |
| Topic Words: | ducks, chickens, farm |
| Equation: | $N_0 * N_1 + N_2 + N_0$ ( $23 * 2 + 6 + 23$ ) |
| Final Answer: | 75 |

In the MWP solving problem, the input is the MWP and the output is the equation or the final answer. In the MWPG problem, the input is the topic words and equation and the output is the MWP.

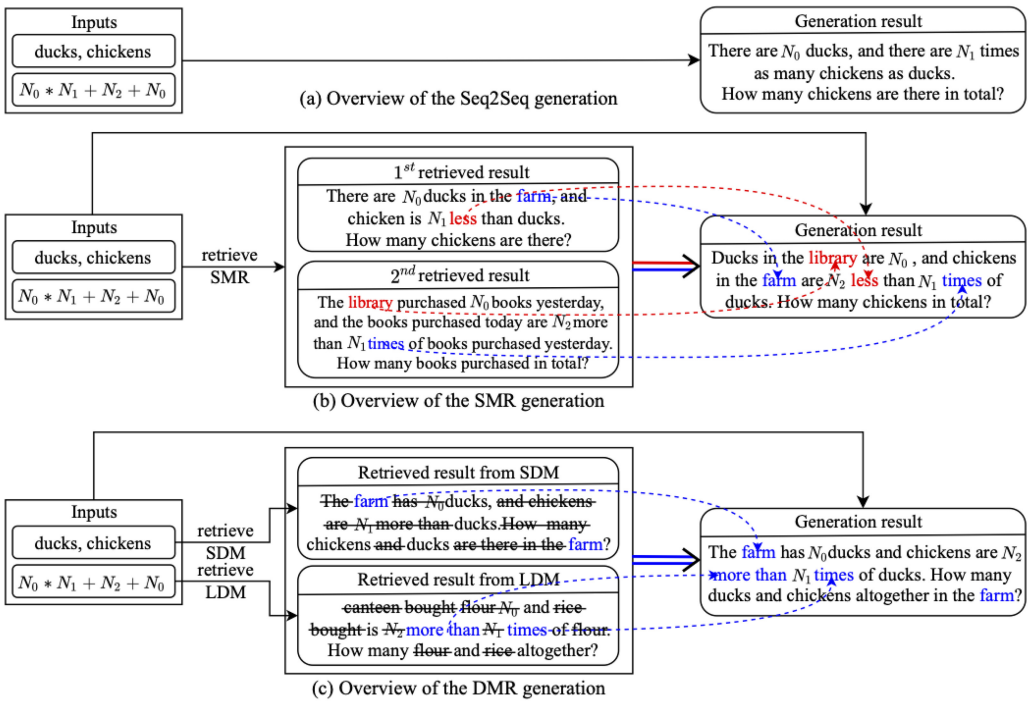


Fig. 1. Illustration about the generation of Seq2Seq, SMR, and DMR. Blue entities represent matching keywords to the generation and red entities represent mismatched keywords to the generation. The crossed words in (c) mean that those words are masked in the pre-processing stage.

inference, we leverage the input topic words and the input equation to retrieve SDM and LDM, respectively. Via this disentangled strategy, the retrieved results could preserve the matching information and avoid the mismatched content. Both the retrieved results are used to augment the generation condition. As shown in Figure 1(c), the input topic words *ducks* and *chicken* retrieve *farm* from SDM, improving the richness of the generated MWP's scenario description. The equation $N_0 * N_1 + N_2 + N_0$ retrieves *more than ... times* from the LDM, helping the generated MWP to be consistent with the input equation.

The contributions are as follows:

— This paper introduces the memory module into the **math word problem generation (MWPG)** task and provides a more comprehensive analysis of the SMR method, including its retrieval quality, training efficiency, and inherent limitations.

— Inspired by the observation that the entanglement of scenario description and logical description will introduce the noisy retrieval results, we propose a novel disentangled memory retrieval framework to augment the input of the generation module with only matching information but not the mismatched information.
— The SMR and DMR outperform all existing MWPG methods. Detailed analysis and discussion verify the effectiveness of the disentangled memory module.

## 2 RELATED WORK

**Math Word Problem Generation.** Traditional methods usually heuristically generate MWPs, based on some pre-defined text templates [7, 41, 45, 61]. However, due to the limited structure of the pre-defined text templates, the MWPs generated by those methods are dull. Recently, deep learning methods have been utilized to enhance the quality of MWP generation. MaGNET [69] is a seq2seq encoder-decoder model that aligns the entities in the generated MWPs with the corresponding variables in input equations. However, this model is limited by the input topic words and equations, and its input to the decoder is restricted. To overcome this limitation, our method, which is also based on a seq2seq encoder-decoder architecture, introduces retrieval modules to retrieve related training data, thus augmenting the input of the decoder. Another recent work [36] employs commonsense knowledge to improve the richness of MWP generation. Wang et al. Reference [59] introduced a GPT-based method to ensure that the equation that solves the generated MWP is the same as the input equation. Compared to our methods, GPT-based models require significantly more computational resources, which will be analyzed in Section 5.1. However, due to its ability to better encode text, we aim to investigate the integration of retrieval modules with GPT models in future work. As noted in Reference [36], the generation quality is impacted by the insufficient conditions of input topic words and equations only.

**Math Word Problem Solver.** There also are some works using deep models to automatically solve the **math word problems (MWPs)** [55, 57]. Tree structure decoders are introduced to boost the generation results [34, 65]. Several works leverage graph neural networks to encode the math word problems [26, 62]. Recently, pre-trained language models are employed as the math word problem encoders [28, 29].

**Text Generation and Memory Retrieval.** Deep neural networks including large pre-training language models have been shown to implicitly store the knowledge in their parameters [31, 46, 56, 64, 68]. To utilize the knowledge in a more explicit and interpretable way, memory retrieval modules are introduced [30, 38–40, 49]. Retrieval modules are shown to bring significant improvement in a number of natural language processing tasks especially the text generation tasks. Some recent works [4, 18, 23] leverage the retrieval results to improve the quality of the generated answers for open-domain question answering. In the dialogue response generation tasks, References [1, 2, 27, 60, 63] leverage the results of retrieval systems to generate more informative and diverse responses. To boost the machine translation quality, Reference [3] copies the retrieved target language sentences to the generated sentences via the cross attention mechanism. This work verifies the effectiveness of the retrieval system in the cross-lingual setting. Retrieval systems are also employed to augment the language pre-training models, which allows the models to retrieve documents from a large corpus and to generate more informative text [10, 11, 19]. In the code generation task, the introduced retrieval system attends over all the available code repositories and the retrieved codes are used to augment the input of the generation modules [12, 14]. Experiments in Reference [25] demonstrate that the retrieval-augmented generation could improve the performance of several knowledge-intensive natural language processing tasks. Inspired by those works, we introduce the retrieval module for the MWPG task and use the retrieved results as an extra generation condition beside the input.

**Disentanglement.** Recent AI research has emphasized the importance of learning disentangled representations of data. To quantitatively evaluate the disentangled representations, References [5, 8] propose a framework for the quantitative evaluation of disentangled representations. Some recent work use the disentanglement strategy to boost the performance of different specific tasks. For the pretrained vision-language model, Reference [33] finds that the generated vision-language representations are entangled in one common latent space and then proposed a disentangled framework that applies explicitly separated attention spaces for vision and language. To solve the MWP, Reference [15] divides the question into two parts, i.e., a concept representation that captures its explicit concept meaning and an individual representation that preserves its personal characteristics. Reference [44] presents a non-parametric algorithms for symmetry-based disentangling of data manifolds. Reference [37] rethinks several commonly held assumptions in the disentangled representations and releases a new library to train and evaluate disentangled representations. A recent work [66] leverages the disentangled strategy to implement the causal intervention in video moment retrieval task.

## 3 PROBLEM SETUP

Following Reference [59], we formulate MWP generation as a task of multi-view (topic words and an equation) conditional text generation. Specifically, we feed the generation network $p_\Theta$ with topic words $x_i^{tw}$ and the equation $x_i^{eq}$, and the output is the generated MWP $\hat{M}_i$. The generated MWP $\hat{M}_i$ is expected to be same as the generation target $M_i$ and consistent with the input equation $x_i^{eq}$ (the detailed evaluation metric will be discussed in Section 5). Then, we describe the MWP generation process as

$$\hat{M}_i = p_\Theta\left(x_i^{eq}, x_i^{tw}\right), \tag{1}$$

where $\{M_i, x_i^{eq}, x_i^{tw}\}$ is the $i$th example in the dataset; $M_i = \{m_1, \ldots, m_T\}$, as the generation target, represents the MWP as a sequence of $T$ tokens. Similarly, $\hat{M}_i = \{\hat{m_1}, ..., \hat{m_{T'}}\}$ represents the generated MWP as a sequence of $T$ generated tokens.

## 4 MEMORY RETRIEVED-BASED MWP GENERATION FRAMEWORK

### 4.1 Overview of our Memory Retrieval-based Approach

The MWPG framework based on memory retrieval is composed of pre-processing stage, retrieval module, and generation module.

**Pre-processing.** Before training, we build the memory $\Phi$ by recording all training MWPs. During inference, the retrieval module will retrieve the built memory.

**Retrieval Module.** In this module, we use the query $q$ to retrieve the memory $\Phi$ to obtain the top $N$ relevant memory items, according to the relevance score $f(q, \varphi_j)$. We define the relevance score $f(q, \varphi_j)$ between the query $q$ and the memory item $\varphi_j$ as the inner product of their representation,

$$f(q, \varphi_j) = ENC_q(q)^T ENC_\varphi(\varphi_j), \tag{2}$$

where the specific query $q$, $\varphi$ and encoders are to be determined in the specific implementation. And then the retrieved results will be fed into the generation module with the original input.

**Generation Module.** This module is a common encoder-decoder generation framework. The original input (topic words and equation) and the retrieved results are fed into the generation module as its condition. By encoding and decoding those condition, the generation module outputs the generated MWPs.

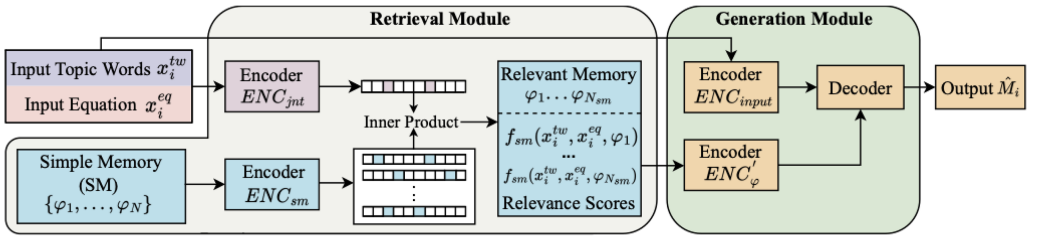


Fig. 2. Framework of our SMR consists of the retrieval module and the generation module. In the retrieval module, we use the input topic words $x_i^{tw}$ and the input equation $x_i^{eq}$ as a joint query to retrieve the SM $\Phi$. According to the relevant score (i.e., the inner product between the presentations of the joint query and the memory item), we select top N retrieved items. Conditioned on the retrieved items and the input, the generation module outputs the MWP.

### 4.2 Simple Memory Retrieval

Based on the overview of Section 4.1, the proposed SMR also contains the Pre-processing stage and the Memory Retrieval Module (i.e., the Simple Memory Retrieval Module) and its specific Generation Module. In further, SMR determines the specific implementation of all the encoders, decoders, and the query. The framework of the proposed SMR is shown in Figure 2.

**Pre-processing.** We build the SM $\Phi$ by recording each training MWP $\{M_i\}$ as a memory item $\varphi_j$.

**Simple Memory Retrieval Module.** In this module, we use the joint query (both the topic words $x_i^{tw}$ and equation $x_i^{eq}$) to retrieve SM $\Phi$ and obtain the top $N_{sm}$ relevant memory items $\{\varphi_j\}_{j=0}^{N_{sm}}$. Since the memory is constructed by recording the MWPs of all the training examples, it is important to note that the input for each example will not retrieve its corresponding target MWP from the memory. The relevance score $f_{sm}(x_i^{tw}, x_i^{eq}, \varphi_j)$ is defined as

$$f_{sm}\left(x_i^{tw}, x_i^{eq}, \varphi_j\right) = ENC_{jnt}\left(x_i^{tw}, x_i^{eq}\right)^T ENC_{sm}(\varphi_j), \tag{3}$$

where $ENC_{jnt}$ and $ENC_{sm}$ encode the query and the SM item and are defined as

$$ENC_{jnt}\left(x_i^{tw}, x_i^{eq}\right) = \delta\left(W_{jnt}\left(Tr_{tw}\left(x_i^{tw}\right) + GRU\left(x_i^{eq}\right)\right)\right), \tag{4}$$

$$ENC_{sm}(\varphi_j) = \delta(W_{sm} Tr_{sm}(\varphi_j)), \tag{5}$$

where $Tr_{tw}$ and $Tr_{sm}$ are the Transformer [51] encoder of the input topic words $x_i^{tw}$ and the SM item $\varphi_j$, respectively. *GRU* stands for Gated Recurrent Units, which are commonly utilized to decode sequential information. In this equation, we use *GRU* to decode the input equation. $W_{jnt}$ and $W_{sm}$ are the matrices of the linear projections, which reduce the dimension of the representations. Function $\delta()$ could normalize any vector to a unit vector, regulating the range of the relevance score.

**Generation Module.** Conditioned on both the original input $(x_i^{tw}, x_i^{eq})$ and the retrieved results $\{\varphi_j\}_{j=1}^{N_{sm}}$ from the retrieval module, our generation module, built upon standard encoder-decoder structure, outputs the generated MWP $\hat{M}_i$. Therefore, the generation module could be regarded as a probabilistic model,

$$p\left(\hat{M}_i | x_i^{tw}, x_i^{eq}, \varphi_1, \ldots, \varphi_{N_{sm}}, f_{sm}\left(x_i^{tw}, x_i^{eq}, \varphi_1\right), \ldots, f_{sm}\left(x_i^{tw}, x_i^{eq}, \varphi_{N_{sm}}\right)\right). \tag{6}$$

We use the retrieved memory to augment the generation module by copying them into generation via the cross attention mechanism [50]. The cross attention mechanism copies related tokens in the retrieved MWP to the generation outputs.

*The encoder* encodes the original input $(x_i^{tw}, x_i^{eq})$ and every retrieved memory item (i.e., MWP) $\varphi_j$ into representations,

$$v_i^{input} = ENC_{input}\left(x_i^{tw}, x_i^{eq}\right), \tag{7}$$

$$v_{\varphi_j} = ENC_{\varphi}^{'}(\varphi_j), \tag{8}$$

where the functions $ENC_{input}$ and $ENC_{\varphi}^{'}$ are similar to $ENC_{jnt}$ and $ENC_{sm}$ defined in Equations (3) and (4), respectively. In Equation (7), the $ENC_{input}$ encodes the input $x_i^{tw}$ and $x_i^{eq}$ into the representation $v_i^{input}$. In Equation (8), the $ENC_{\varphi}^{'}$ encodes each retrieved memory item (i.e., MWP) $\varphi_j$ into the representation $v_{\varphi_j}$ individually, resulting in a set of contextualized token embeddings $\{v_{\varphi_j k}\}_{k=1}^{L_j}$, where $L_j$ denotes the length of the token sequence $\varphi_j$.

*The decoder* can be regarded as a probabilistic model,

$$p\left(\hat{M}_i | v_i^{input}, v_{\varphi_1}, \ldots, v_{\varphi_{N_{sm}}}, f_{sm}\left(x_i^{tw}, x_i^{eq}, \varphi_1\right), \ldots, f_{sm}\left(x_i^{tw}, x_i^{eq}, \varphi_{N_{sm}}\right)\right). \tag{9}$$

Fed with the presentations $v_i^{input}$ and $\{v_{\varphi_j}\}_{j=1}^{N_{sm}}$, the decoder generates an output sequence $\hat{M}_i$ in an auto-regressive fashion. At each timestep $t$, the generation decoder attends over both the representation $v_i^{input}$ from the encoder and previously predicted sequence $\hat{m}_{1:t-1}$, outputting a hidden state $h_t$. The hidden state $h_t$ is then converted to next-token probabilities through a linear projection followed by softmax function,

$$P_v = softmax(W_v h_t + b_v). \tag{10}$$

In addition, we compute a cross attention over the representations of all retrieved memory items,

$$\alpha_{jk} = \frac{exp(h_t^T W_m v_{\varphi_j k})}{\sum_{j=1}^{N_{sm}} \sum_{k=1}^{L_j} exp(h_t^T W_m v_{\varphi_j k})}, \tag{11}$$

$$c_t = W_c \sum_{j=1}^{N_{sm}} \sum_{k=1}^{L_j} \alpha_{jk} v_{\varphi_j k}, \tag{12}$$

where $v_{\varphi_j k}$ is the $k$th token in the $j$th retrieved memory, $\alpha_{jk}$ is the attention score of $v_{\varphi_j k}$, $c_t$ is a weighted combination of memory embeddings, and $W_m$ and $W_c$ are trainable matrices. The cross attention is used twice in the decoding stage. First, we update the decoder's hidden state by a weighted sum of memory embeddings, i.e., $h_t = h_t + c_t$. Second, we regard every cross attention score as a probability of copying the corresponding token of the retrieved memory items [9, 50]. Therefore, we use $P_v$ and the weighted combination of memory embeddings $c_t$ to compute the final next-token probabilities as

$$p(\hat{m}_t|\cdot) = (1 - \lambda_t) P_v(\hat{m}_t) + \lambda_t \sum_{j=1}^{N_{sm}} \sum_{k=1}^{L_j} \alpha_{jk} \mathbb{1}_{v_{\varphi_j k} = \hat{m}_t}, \tag{13}$$

where $\mathbb{1}$ is the indicator function and $\lambda_t$ is a gating variable computed by another feed-forward network $\lambda_t = g(h_t, c_t)$.

Inspired by References [3, 24], to enable the gradient flow from the generation output to the simple retrieval module, we add the relevance scores as bias items on the attention scores, so we rewrite Equation (11) as

$$\alpha_{jk} = \frac{exp(h_t^T W_m v_{\varphi_j k} + \beta f_{sm}(x_i^{tw}, x_i^{eq}, \varphi_j))}{\sum_{j=1}^{N_{sm}} \sum_{k=1}^{L_j} exp(h_t^T W_m v_{\varphi_j k} + \beta f_{sm}(x_i^{tw}, x_i^{eq}, \varphi_j))}, \tag{14}$$

where $\beta$ is a trainable scalar that controls the weight of the relevance scores.

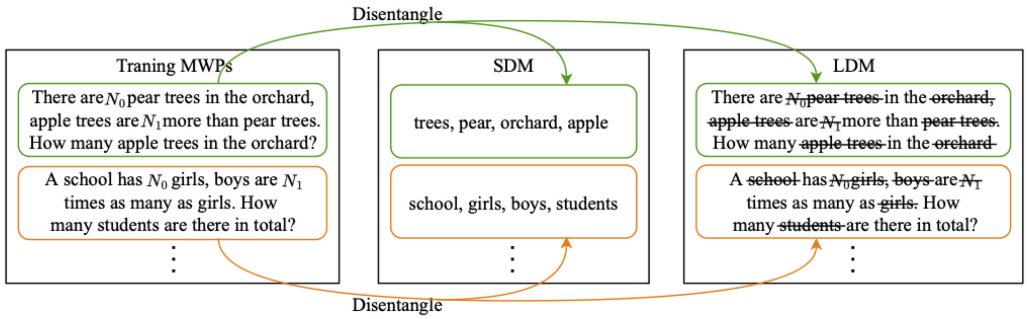


Fig. 3. The pre-processing of DMR. We first disentangle each training MWP and then use them to build SDM and LDM.

### 4.3 Disentangled Memory Retrieval

Similarly to the SMR, our DMR also contains the Pre-processing stage and Memory Retrieval Module (i.e., the Disentangled Memory Retrieval Module) and its specific Generation Module. Figure 3 illustrates that, in contrast to the SMR, our DMR approach separates the training MWP during the pre-processing phase and establishes two distinct memory structures. Our DMR Module consists of two parallel retrieval modules where we use different input as queries to retrieved the two disentangled memories, respectively. All the retrieved results and the input are fed into its specific generation module that output the generated MWP. The framework of the proposed DMR is shown in Figure 4.

**Pre-processing.** In the pre-processing stage, as shown in Figure 3, we *disentangle training MWPs* $\{M_i\}_{i=1}^N$ into logical description $\{M_i^{ld}\}_{i=1}^N$ and scenario description $\{M_i^{sd}\}_{i=1}^N$ and *build the memories*, i.e., LDM and SDM. Initially, we utilize dependency parsing technology to analyze the sentences in the MWP. Subsequently, we identify the root verb and additional verbs that exhibit a coordination relationship with the root verb. In the dependency tree, all child nodes that have relationships of nsubj, obj, iobj, or obl with the above verb nodes, along with their respective subtrees, are considered to be part of the scenario description. The verb nodes and all the child nodes that are related to the verb nodes via advmod, advcl, or mark relationships, as well as their corresponding subtrees, are considered part of the logical description. Interrogative words are also considered part of the logical description. The numbers are discarded. Further, Figure 3 shows that we record all logical description $\{M_i^{ld}\}_{i=1}^N$ and scenario description $\{M_i^{sd}\}_{i=1}^N$ into LDM $\Phi^{ldm}$ and SDM $\Phi^{sdm}$, respectively.

**Disentangled Retrieved Module.** This module contains two independent retrieved modules, i.e., *topic-words-based retrieved module* and *equation-based retrieved module*. Each of them is a complete retrieval module like SMR.

*In Topic-words-based retrieved module*, we use the topic words $x_i^{tw}$ as the query to retrieve SDM $\Phi^{sdm}$ and obtain the top $N_{sdm}$ relevant SDM items $\{\varphi_j^{sdm}\}_{j=0}^{N_{sdm}}$. The relevant score $f_{tw}(x_i^{tw}, \varphi_j^{sdm})$ is defined as

$$f_{tw}\left(x_i^{tw}, \varphi_j^{sdm}\right) = ENC_{tw}\left(x_i^{tw}\right)^T ENC_{sdm}\left(\varphi_j^{sdm}\right) \tag{15}$$

and the encoders are defined as

$$ENC_{tw}\left(x_i^{tw}\right) = \delta\left(W_{tw} Tr_{tw}\left(x_i^{tw}\right)\right), \tag{16}$$

$$ENC_{sdm}\left(\varphi_j^{sdm}\right) = \delta\left(W_{sd} Tr_{cn}\left(\varphi_j^{sdm}\right)\right), \tag{17}$$

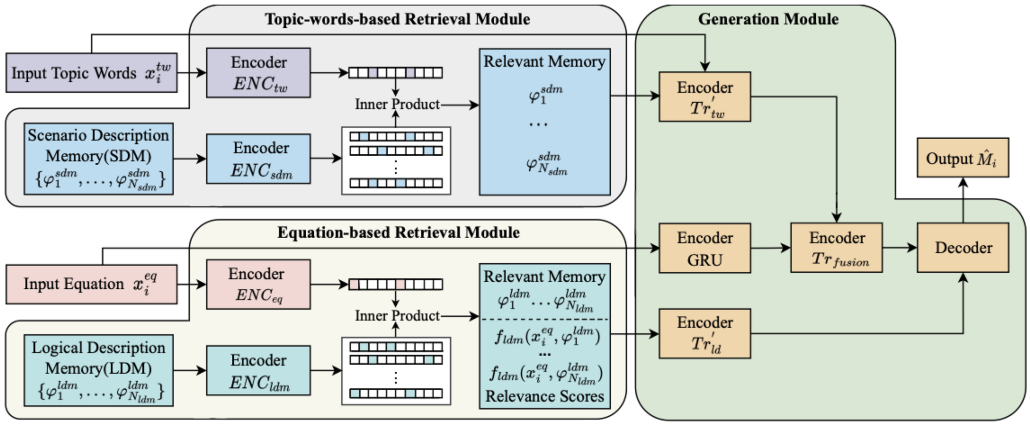


Fig. 4. Framework of our DMR consists of the topic-words-based retrieval module, the equation-based retrieval module, and the generation module. In the topic-words-based retrieval module, we use the input topic words to retrieve the SDM. According to the relevant score (i.e., the inner product between representations of the input topic words and SDM item), we select top N retrieved SDM items. Similarly, in the equation-based retrieval module, we use the input equation to retrieve the LDM. According to the inner product between the representations of the input equation and the LDM item, we select top N retrieved LDM items. The retrieved results from both the LDM and the SDM are used to augment the input of the generation module, which generates the MWP.

where $Tr_{tw}$ and $Tr_{sdm}$ are the Transformer encoders for the input topic words $x_i^{tw}$ and the retrieved SDM items $\varphi_j^{sdm}$. $W_{tw}$ and $W_{sdm}$ are the matrices of the linear projections, which reduce the dimension of the representations. Function $\delta()$ could normalize any vector to a unit vector, regulating the range of the relevance score.

*In equation-based retrieved module*, we use the input equation $x_i^{eq}$ as the query to retrieve LDM $\Phi_{eq}$ and obtain the top $N_{ldm}$ relevant LDM items $\{\varphi_j^{ldm}\}_{j=0}^{N_{ldm}}$. The relevant score $f_{eq}(x_i^{eq}, \varphi_j^{ldm})$ is defined as

$$f_{eq}\left(x_i^{eq}, \varphi_j^{ldm}\right) = ENC_{eq}\left(x_i^{eq}\right)^T ENC_{ldm}\left(\varphi_j^{ldm}\right), \tag{18}$$

$$ENC_{eq}\left(x_i^{eq}\right) = \delta\left(W_{eq} GRU_{eq}\left(x_i^{eq}\right)\right), \tag{19}$$

$$ENC_{ldm}\left(\varphi_j^{ldm}\right) = \delta\left(W_{ldm} Tr_{ldm}\left(\varphi_j^{ldm}\right)\right), \tag{20}$$

where the function of $GRU_{eq}$, $Tr_{ldm}$, $W_{eq}$, $\delta$, and $W_{ldm}$ are similar to $Tr_{tw}$, $Tr_{sdm}$, $W_{tw}$, $\delta$, and $W_{sdm}$ mentioned in Equation (16) and Equation (17), respectively. In Equation (19), we employ $GRU_{eq}$ rather than Transformer to encode the equation $x_i^{eq}$, since using GRU achieves better performance empirically.

**Generation Module.** Conditioned on both the original input $(x_i^{tw}, x_i^{eq})$ and the retrieved results $(\{\varphi_j^{sdm}\}_{j=1}^{N_{sdm}}, \{\varphi_j^{ldm}\}_{j=1}^{N_{ldm}})$ from the disentangled retrieval module, our generation module outputs the generated MWP $\hat{M}_i$. Therefore, the generation module could be regarded as a probabilistic model,

$$p\left(\hat{M}_i | x_i^{tw}, x_i^{eq}, \varphi_1^{sdm}, \ldots, \varphi_{N_{sdm}}^{sd}, \varphi_1^{ldm}, \ldots, \varphi_{N_{ldm}}^{ldm}, f_{ldm}\left(x_i^{eq}, \varphi_1^{ldm}\right), \ldots, f_{ldm}\left(x_i^{eq}, \varphi_{N_{ldm}}^{ldm}\right)\right). \tag{21}$$

Since the retrieved scenario description $\{\varphi_j^{sdm}\}_{j=1}^{N_{sdm}}$ is a set of nouns without structure information, we use them to augment the input topic words $x_i^{tw}$ directly. On the contrary, since the

retrieved logical description $\{\varphi_j^{ldm}\}_{j=1}^{N_{ldm}}$ contains the structure information, we copy the retrieved logical description into generation via the cross attention mechanism [50]. The generation module consists of an encoder and a decoder.

*The encoder* encodes the original input $(x_i^{tw}, x_i^{eq})$ and the retrieved results $(\{\varphi_j^{sdm}\}_{j=1}^{N_{sdm}}, \{\varphi_j^{ldm}\}_{j=1}^{N_{ldm}})$ into representations,

$$v_i^{tw} = Tr'_{tw}\left(x_i^{tw}, \varphi_1^{sdm}, \ldots, \varphi_{N_{sdm}}^{sdm}\right), \tag{22}$$

$$v_i^{eq} = GRU\left(x_i^{eq}\right), \tag{23}$$

$$v_i^{fs} = Tr_{fusion}\left(v_i^{tw}, v_i^{eq}\right), \tag{24}$$

$$v_{\varphi_j^{ldm}} = Tr'_{ldm}\left(\varphi_j^{ldm}\right), \tag{25}$$

In Equation (22), the Transformer $Tr'_{tw}$ encodes the input topic words $x_i^{tw}$ and all the retrieved SDM items $\{\varphi_j^{sdm}\}_{j=1}^{N_{sdm}}$ into the representation $v_i^{tw}$. In Equation (23), the *GRU* encodes the input equation $x_i^{eq}$ into the representation $v_i^{eq}$. In Equation (24), the Transformer $Tr_{fusion}$ fuses $v_i^{tw}$ and $v_i^{eq}$ into $v_i^{fs}$. In Equation (25), the logical description Transformer encoder $Tr'_{ldm}$ encodes each the retrieved LDM item $\varphi_j^{ldm}$ individually, resulting in a set of contextualized token embeddings $\{v_{\varphi_j k}\}_{k=1}^{L_j}$, where $L_j$ denotes the length of the token sequence $\varphi_j^{ldm}$.

*The decoder* can be regarded as a probabilistic model,

$$p\left(M_i | v_i^{fs}, v_{\varphi_1^{ldm}}, \ldots, v_{\varphi_{N_{ldm}}^{ldm}}, f_{ldm}\left(x_i^{eq}, \varphi_1^{ldm}\right), \ldots, f_{ldm}\left(x_i^{eq}, \varphi_{N_{ldm}}^{ldm}\right)\right). \tag{26}$$

Fed with the presentations $v_i^{fs}$, $\{v_{\varphi_j^{ldm}}\}_{j=1}^{N_{ldm}}$ and corresponding relevance score$\{f_{ldm}(x_i^{eq}, \varphi_j^{ldm})\}_{j=1}^{N_{ldm}}$, the decoder generates an output sequence $\hat{M}_i$ in an auto-regressive fashion. At each timestep $t$, the generation decoder attends over both the representation $v_i^{fs}$ from the encoder and previously predicted sequence $m_{1:t-1}$, outputting a hidden state $h_t$. The hidden state $h_t$ is then converted to next-token probabilities through a linear projection followed by softmax function,

$$P_v = softmax(W_v h_t + b_v). \tag{27}$$

In addition, we also compute a cross attention over the representation of all tokens of all retrieved LDM items,

$$\alpha_{jk} = \frac{exp(h_t^T W_m v_{\varphi_j^{ldm}k})}{\sum_{j=1}^{N_{ldm}} \sum_{k=1}^{L_j} exp(h_t^T W_m v_{\varphi_j^{ldm}k})}, \tag{28}$$

$$c_t = W_c \sum_{j=1}^{N_{ldm}} \sum_{k=1}^{L_j} \alpha_{jk} v_{\varphi_j^{ldm}k}, \tag{29}$$

where $\alpha_{jk}$ is the attention score of the $k$th token in the $j$th retrieved LDM item, $c_t$ is a weighted combination of memory embeddings, and $W_m$, $W_c$ are trainable matrics. Similarly to the SMR, the cross attention is used twice in the decoding stage. First, we update the decoder's hidden state by a weighted sum of memory embeddings, i.e., $h_t = h_t + c_t$. Second, we regard every cross attention score as a probability of copying the corresponding token of the retrieved memory items. Therefore, we use $P_v$ and the weighted combination of memory embeddings $c_t$ to compute the final

next-token probabilities are computed as

$$p(\hat{m}_t|\cdot) = (1-\lambda_t)P_v(\hat{m}_t) + \lambda_t \sum_{j=1}^{N_{ldm}} \sum_{k=1}^{L_i} \alpha_{jk} \mathbb{1}_{v_{\varphi_j^{ldm}{}_k}=\hat{m}_t}, \tag{30}$$

where $\mathbb{1}$ is the indicator function and $\lambda_t$ is a gating variable computed by another feed-forward network $\lambda_t = g(h_t, c_t)$.

Similarly to *SMR*, we add the LDM relevance scores as bias items on the attention scores, so we rewrite Equation (28) as

$$\alpha_{jk} = \frac{exp(h_t^T W_m v_{\varphi_j k} + \beta f_{ldm}(x_i^{eq}, \varphi_j^{ldm}))}{\sum_{j=1}^{N_{ldm}} \sum_{k=1}^{L_j} exp(h_t^T W_m v_{\varphi_j k} + \beta f_{ldm}(x_i^{eq}, \varphi_j^{ldm}))}, \tag{31}$$

where $\beta$ is a trainable scalar that control the weight of the relevance scores. We do not add the SDM relevance scores on the attention score. On the one hand, we pre-train the encoders of the topic-words-based module and its retrieval performance is good enough. On the other hand, the SDM retrieved results are used to augment the input topic words directly rather than copied by the cross attention mechanism. Technically, we cannot add the SDM relevance to the cross attention of LDM.

### 4.4 Training

We optimize the parameters $\Theta$ of the model using stochastic gradient descent on the negative log − likelihood loss function,

$$\mathcal{L} = -\log p\left(M_i|x_i^{tw}, x_i^{eq}, \varphi_1^{sdm}, \ldots, \varphi_{N_{sdm}}^{sdm}, \varphi_1^{ldm}, \ldots, \varphi_{N_{ldm}}^{ldm}, f_{ldm}\left(x_i^{eq}, \varphi_1^{ldm}\right), \ldots, f_{ldm}\left(x_i^{eq}, \varphi_{N_{ldm}}^{ldm}\right)\right), \tag{32}$$

where $M_i$ refers to the target MWP. To improve training efficiency, we warm-start the retrieval module by pre-training the four encoders in the disentangled retrieval module with a cross-alignment task.

**Pre-training for topic-words-based retrieval module.** We sample all topic-words and scenario description pairs $\{x_i^{tw}, \varphi_i^{sdm}\}_{i=1}^N$ from training set and SDM at each training step. Let $X_{tw} \in R^{B\times b}$ and $P_{sdm} \in R^{B\times b}$ be the representation of the topic words and scenario description through $ENC_{tw}$ and $ENC_{sdm}$, respectively. $S = X_{tw}P_{sdm}^T$ is a $(B \times B)$ matrix of relevance scores, where each row corresponds to the topic words of one training example and each column corresponds to the scenario description of one SDM item. Any $(X_{tw,i}, P_{sdm,j})$ pairs should be aligned when $i = j$ and should not otherwise. Therefore, the loss function should maximize the scores along the diagonal of the matrix and minimize the other scores. The loss function can be written as

$$\mathcal{L}_{tw}^{(i)} = \frac{-exp(S_{ii})}{exp(S_{ii}) + \sum_{j\neq i} exp(S_{ij})}. \tag{33}$$

**Pre-training for equation-based retrieval module.** We sample all equation and logical-description pairs $\{x_i^{eq}, \varphi_i^{ldm}\}_{i=1}^N\}$ from the training set and LDM at each training step. Let $X_{eq} \in R^{B\times b}$ and $P_{ldm} \in R^{B\times b}$ be the representation of the equation and logical description through $ENC_{eq}$ and $ENC_{ldm}$, respectively. Similarly to $S$ in Equation (33), $U = X_{eq}P_{ldm}^T$ is a $(B \times B)$ matrix of relevance scores between the equation and retrieved logical description from LDM. Thus, the loss for this module is computed as follows:

$$\mathcal{L}_{eq}^{(i)} = \frac{-exp(U_{ii})}{exp(U_{ii}) + \sum_{j\neq i} exp(U_{ij})}. \tag{34}$$

Table 2. Summary Statistics of Datasets

| | #trainset | #valset | #testset | total |
|---|---|---|---|---|
| Math23K | 16,781 | 2,083 | 2,111 | 20,975 |
| Dolphin18K | 7,593 | 847 | 2,110 | 10,550 |
| MAWPS | 1,865 | 241 | 241 | 2,347 |

## 5 EXPERIMENTS

We now perform a series of experiments to validate the effectiveness of our proposed MWP generation approach.

**Datasets.** We perform experiments on three commonly used MWP solving datasets, i.e., Math23K [58], MAWPS [20], and Dolphine18K [13]. Following the splitting strategy of Reference [21], we split each dataset into train set, validation set, and test set. The summary statistics of datasets are shown in Table 2.

To transfer those MWP solving datasets into MWPG datasets, we obtain equation and topic words for each problem as their input. We extract as most $n_{tp}$ words with highest TF-IDF scores as the topic words in our experiments. As shown in Table 1, the equation $N_0 * N_1 + N_2 + N_0$ and the extracted topic words *ducks, chickens* is the input and the MWP is its ground-truth label. For a fair comparison, we follow the settings of baselines and set $n_{tp} = 5$, $n_{tp} = 10$, and $n_{tp} = 5$ on Math23K, Dolphin18K, and MAWPS, respectively. Different from Math23K and MAWPS, Dolphin18K is a multiple-equation MWP dataset. Following Reference [69], we concatenate multiple equations as a single equation.

**Baselines.** In Table 3, *seq2seq-rnn*, based on the LSTMs with attention [35, 69], regards the MWP generation task as a sequence-to-sequence task, which splices the input equation and the input topic words together as a single sequence input. Compared with *seq2seq-rnn*, *seq2seq-rnn-glove* uses GloVe [43] instead of random embeddings at initialization and *seq2seq-tf* is based on Transformers [51] rather than RNN. We also compare our approach to vanilla GPT-2 [47], either with fine-tuning or not; we denote these models as *GPT* and *GPT-ft*, respectively. Based on *GPT-ft*, *MCPCC* introduces an equation consistency constraint, which encourages the generated MWP to contain the exact same equation as the one used to generate it [59]. In Table 4, *MaGNET* [69], based on a standard seq2seq encoder-decoder architecture, forces the entities in the generated MWP to correspond to the variables in the equation. *KNN*, *Equ2Math*, and *Topic2Math* are MaGNET's ablation methods. In the original papers of baselines [59, 69], experiments are only performed on part of those three datasets. Therefore, our method is compared with different baselines on different datatsets.

**Metrics.** We leverage the following three commonly used evaluation metrics: BLEU-4 [42], **Metric for Evaluation of Translation with Explicit ORdering ()METEOR** [22], and **Recall-Oriented Understudy for Gisting Evaluation (ROUGE)** [32] to measure the language quality. **BiLingual Evaluation Understudy (BLEU)**, METEOR, and ROUGE are proposed to evaluate the quality of machine translation. Then they are used as evaluation metrics in many text generation tasks, e.g., machine translation, image caption and dialog generation. In the BLEU evaluation metric, BLEU-4 refers to the use of a modified 4-gram precision function. The BLEU score is commonly used to measure the level of correspondence between the generated text and the ground-truth text. The METEOR score measures the similarity between the generated text and the target text by computing the harmonic mean of precision and recall, where recall is given more weight than precision. It is based on a generalized concept of unigram matching, which considers unigrams in the generated and target text, matching them based on their surface form, stemmed form, or meaning. ROUGE and BLEU are both commonly used evaluation metrics in text generation tasks;

Table 3. Experiment Results on MAWPS and Math23k

| | MAWPS | | | | Math23K | | | |
|---|---|---|---|---|---|---|---|---|
| | BLEU-4 | METEOR | ROUGE-L | ACC-eq | BLEU-4 | METEOR | ROUGE-L | ACC-eq |
| Seq2Seq-rnn | 0.153 | 0.175 | 0.362 | 0.472 | 0.196 | 0.234 | 0.444 | 0.390 |
| +GloVe | 0.592 | 0.412 | 0.705 | 0.585 | 0.275 | 0.277 | 0.507 | 0.438 |
| Seq2Seq-tf | 0.544 | 0.387 | 0.663 | 0.588 | 0.301 | 0.294 | 0.524 | 0.509 |
| GPT | 0.368 | 0.294 | 0.538 | 0.532 | 0.282 | 0.297 | 0.512 | 0.477 |
| GPT-ft | 0.504 | 0.391 | 0.664 | 0.512 | 0.325 | 0.333 | 0.548 | 0.498 |
| MCPCC | 0.596 | 0.427 | 0.715 | 0.557 | 0.329 | 0.328 | 0.544 | 0.505 |
| SMR | 0.618 | 0.557 | 0.741 | 0.590 | 0.330 | 0.333 | 0.545 | 0.511 |
| DMR | **0.655** | **0.610** | **0.778** | **0.618** | **0.399** | **0.377** | **0.637** | **0.544** |

Bold numbers indicate the best results.

Table 4. Experiment Results on Dolphin18K

| | BLEU-4 | METEOR | ROUGE-L |
|---|---|---|---|
| Equ2Math | 0.050 | 0.135 | 0.296 |
| KNN | 0.120 | 0.168 | 0.361 |
| Topic2Math | 0.123 | 0.239 | 0.422 |
| MaGNET | 0.125 | 0.248 | 0.436 |
| SMR | 0.158 | 0.312 | 0.407 |
| DMR | **0.238** | **0.367** | **0.496** |

Bold numbers indicate the best results.

however, they differ in their focus. BLEU is based on precision, while ROUGE is based entirely on recall. For a comprehensive understanding of the metrics, refer to Reference [42] for BLEU, [22] for METEOR, and [32] for ROUGE. We implement those three metrics using the package provided in Reference [6]. For mathematical consistency, we use the equation accuracy (ACC-eq) metric that measures whether the generated MWP is mathematically consistent with the input equation.

### 5.1 Quantitative Results

**Comparsion with baselines.** We show the quantitative results of our experiments performed on MAWPS, Math23K, and Dolphin18K in the Table 3 and Table 4. As shown in Table 3, both our SMR and DMR get higher BLEU-4, METEOR, and ROUGE-L scores on the MAWPS and Math23K. Therefore, the MWPs generated by our methods have better language quality than the MWPs generated by both the seq2seq-based methods and GPT-based methods (i.e., GPT, GPT-ft, and MCPCC). SMR and DMR also have higher Acc-eq scores, which means that the generated MWPs of our methods are more consistent with the input equation. Thus, our method could generated better MWPs than all baselines. However, the metric ACC-eq of the MWPs generated by all the methods including our methods is not good enough. As the best method, the ACC-eq of our DMR achieves only 60.5% and 54.5% on the MAWPS and Math23K, respectively. In other words, about 39.5% and 45.5% of the generated MWPs are not consistent with their input equations and those generated MWPs are meaningless in the real education scenarios. Therefore, there is still a lot of room for improvement in this task.

Table 4 shows our *DMR* outperforms all the baselines on Dolphin18K. The performance on Dolphin18K is worse than the performance on MAWPS and Math23k, since Dolphin18K is a multiple-equation MWP dataset. Compared with generating MWPs on a single-equation dataset, generating MWPs on a multiple-equation MWP datset is much more difficult. Due to the difficulty of

Table 5. The Training Efficiency of Our Methods and Baselines

| Models | #params | training epochs |
|---|---|---|
| seq2seq-rnn | 11M | 5,000 |
| seq2seq-tf | 52M | 5,000 |
| GPT | 774M | 15,000 |
| MCPCC | 774M | 16,000 |
| SMR | 53M | 8,000 |
| DMR | 59.39M | 8,000 |

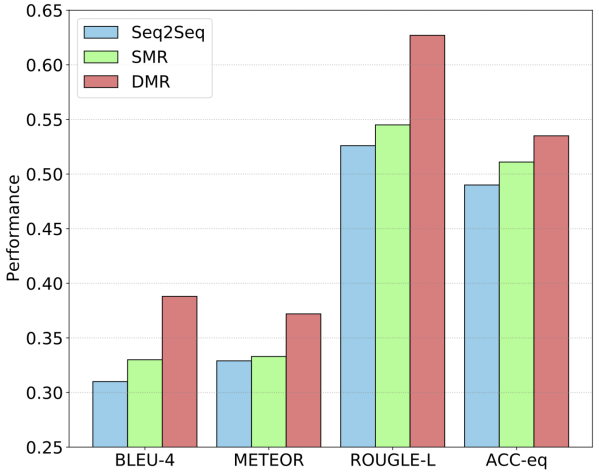


Fig. 5. Ablation study on Math23K.

calculating the accuracy of a multiple-equation MWP, we do not compare the metric ACC-eq on Dolphin18K.

**Training Efficiency.** As demonstrated in Table 5, our methods (SMR and DMR) based on the seq2seq architecture have comparable GPU memory usage and training time to other seq2seq-based methods. However, Table 5 also indicates that GPT-based methods (GPT and MCPCC) require 13 times more GPU memory and take twice as long to train as our methods. This low training efficiency of GPT-based methods limits their application in real-world scenarios. In conclusion, our methods have a lower resource consumption, similar to seq2seq-based methods, while delivering better performance than GPT-based methods.

**Ablation Study.** We perform two ablation methods on Math23K to verify the effectiveness of the memory module and the disentangle strategy, respectively. *seq2seq(ours)* and *seq2seq(ours) w/ memory* are based on the same encoder-decoder structure as our *DMR*. Different with our *DMR*, *seq2seq(ours)* does not contain the memory module and *seq2seq(ours) w/ memory* employs a single memory module without the disentangle strategy. Since Math23K is the largest dataset of those three datasets, the ablation study is performed on the Math23K.

From Figure 5, we can observe that our proposed *SMR*, the seq2seq architecture with a simple memory, performs slightly better than *seq2seq*. This shows that the retrieved results from the simple memory do improve the quality of the generated MWPs. However, the improvement *SMR* bring is limited. According to the case study in Figure 1, we can speculate that this may be because not all information of the retrieved results from the SM is beneficial. Our *DMR* achieves much

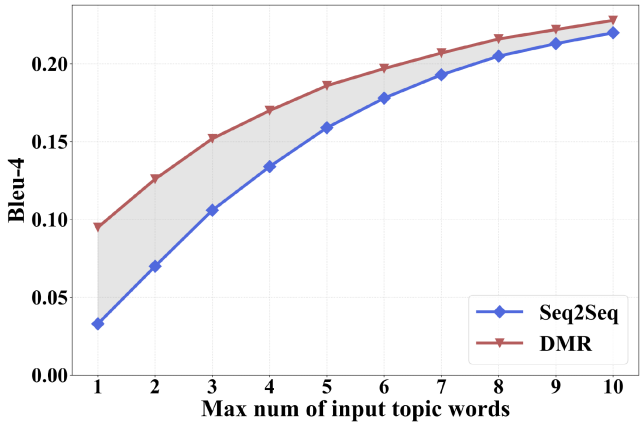


Fig. 6. Experiments with different max numbers of topic words words as input.

better performance than both *seq2seq* and *SMR*. Therefore, we can conclude that the disentangled memory retrieval module is better than the single memory retrieval module.

**Number of the input topic words.** As shown in Figure 6, we perform experiments with different numbers of topic words as input on Dolphin18K. First, with fewer topic words, the generation of MWPs is more difficult. When we decrease the number of the input topic words, the BLEU-4 score is decreasing, too. Only a small number of topic words in the training examples could not fully summarize the scenarios of the MWPs and thus the generation condition is insufficient. Comparing our *DMR* and *Seq2Seq* with different number of input topic words verify that our method could improve the richness of the scenario description. As shown in Figure 6, the fewer topic words we input, the greater gap the sentences generated by our *DMR* and the *Seq2Seq* methods has . Since the retrieved results are leveraged to augment the input of the generation module, our methods, especially the *DMR*, still achieve higher BLEU scores by generating MWPs with rich scene descriptions. Our method can compensate for the problem caused by fewer input topic words. In other words, our *DMR* could improve the richness of the scenario description by augmenting the topic words with retrieved scenario description.

### 5.2 Qualitative Results

**Human Evaluation.** Since automatic evaluation metrics are not always consistent with human judgments on the correctness of a MWP, we conducted human evaluation on our model compared with several baselines mentioned above. We consider three metrics as follows:

— Equation Relevance: whether a problem is relevant to the given equation;
— Topic Word Relevance: whether a problem is relevant to all given topic words;
— Language Fluency: whether a problem is grammatically correct and is fluent to read.

For human evaluation, we randomly selected 100 instances from the Math23K test set and then show the equations and topic words lists with generated math problems from different models to three human annotators to evaluate the generated problems' quality. For each metrics, we ask the annotators to rate the problems on a scale from 1 to 3 (where 3 is the best). Results of each human evaluation metric are presented in Table 6. We can see that our *DMR* has the highest scores across all the metrics. Therefore, the MWPs generated by our method achieve better performance on Equation Relevance, Topic Word Relevance, and Language Fluency.

**Case Study.** From Figure 1 (i.e., the real cases from the generation results of test set), we can observe the intuitive impact brought by *SMR* and *DMR*. From Figure 1(a), the generated

Table 6. Human Evaluation Results

| | Equation Relevance | Topic Words Relevance | Language Fluency |
|---|---|---|---|
| Seq2Seq-tf | 1.71 | 2.34 | 2.19 |
| GPT-pre | 2.17 | 2.57 | 2.55 |
| MCPCC | 2.24 | 2.71 | 2.60 |
| SMR | 2.39 | 2.80 | 2.64 |
| DMR | **2.54** | **2.88** | **2.76** |

Bold numbers indicate the best results.

Table 7. Additional Examples of MWPs Generated by Our Approach

| | |
|---|---|
| **Equation** | $N_0 * N_1$ |
| **Topic words** | car, place |
| **Ground truth** | A car travels $N_0$ kilometers per hour from place A to place B and arrives in $N_1$ hours. How many kilometers are A and B apart? |
| **Gen.MWP** | A car travels $N_0$ kilometers per hour and travels $N_1$ hours from place A to place B in total. How many kilometers is the distance from A to B? |
| **Equation** | $N_0 - N_1$ |
| **Topic words** | group, boys, girls |
| **Ground truth** | There are $N_0$ boys in the group, $N_1$ more than girls. How many girls are there? |
| **Gen.MWP** | There are $N_0$ people in the group, including $N_1$ girls. How many boys are there? |
| **Equation** | $N_0/(N_1 * N_2)$ |
| **Topic words** | library, books, bookshelves, floors |
| **Ground truth** | The library bought $N_0$ books and placed them on $N_1$ bookshelves. Each bookshelf has $N_2$ floors. How many books are on each floor on average? |
| **Gen.MWP** | The library bought $N_0$ books. These books should be placed on $N_1$ bookshelves and each bookshelf is divided into $N_2$ layers. How many books are placed on each layer on average? |
| **Equation:** | $N_0 * N_1 + N_2$ |
| **Topic words** | orchard, trees, apple, peach |
| **Ground truth** | There are $N_0$ rows of apple trees in the orchard, $N_1$ trees in each row, and $N_2$ peach trees. How many trees in the orchard? |
| **Gen.MWP** | There are $N_0$ apple trees in the orchard, and the number of peach trees is $N_2$ more than $N_1$ times the number of apple trees. How many peach trees are there in the orchard? |

We use the blue to mark the novel words and phrases that our method introduce.

MWP of the *seq2seq* is limited to the input topic words. As shown in Figure 1(b), some retrieved results (i.e., "farm" and "times") from the SM of *SMR* facilitate the generation and some accompanying retrieved results (i.e., "library" and "less") damage the generation. Figure 1(c) shows that our *DMR* could only retrieve the beneficial results and avoid the accompanying poisonous results via its disentangled strategy.

Successful examples of MWPs generated by our *DMR* on the test set of Math23K are shown in Table 7. The first generated MWP from Table 7 is almost identical with the ground-truth MWP. In the second case, our DMR introduces a novel topic word "people" corresponding to the input topic words "boys" and "girls." In the third case, our DMR introduces a novel phrase "is divided into" corresponding to division and a novel topic word "layer." In the fourth case, our DMR introduces novel phrases "more than" and "times" corresponding to the addition and multiplication in the input equation.

Table 8. Some Failed Cases Generated by Our Approach

| | |
|---|---|
| **Equation** | $N_0 * N_1 * N_2$ |
| **Topic words** | school, pens |
| **Ground truth** | The school bought $N_0$ boxes of pens, $N_1$ pens per box, $N_2$ yuan each.How much did it cost in total? |
| **Gen.MWP** | The school bought $N_0$ boxes of pens, each with $N_1$ pens, and each pencil cost $N_2$ yuan. How much did it cost in total? |
| **Error** | Redundant topic word (pencil) destroyed the information integrity of the sentence. |
| **Equation** | $N_0 - N_1 - N_2$ |
| **Topic words** | rope |
| **Ground truth** | A rope is $N_0$ meters long, $N_1$ meters are used for the first time, and $N_2$ meters are used for the second time. How many meters are left? |
| **Gen.MWP** | A rope is $N_0$ meters long, $N_1$ meters are used for the first time, and $N_2$ meters are used for the second time. How many meters are used in total? |
| **Error** | Unsuitable query causes equation template to be irrelevant to the input ones. |
| **Equation** | $(N_0 + N_1) * N_2$ |
| **Topic words** | orchard, rows, apple, peach, trees |
| **Ground truth** | In the orchard there were $N_0$ rows of apple trees and $N_1$ rows of peach trees, each rows has $N_2$ trees.,How many trees in this orchard? |
| **Gen.MWP** | There are $N_0$ peach trees in the orchard, which is $N_1$ more than the apple trees. The apple trees are $N_2$ times as many as the peach trees. How many apple trees are there? |
| **Error** | Sentence are incoherent and cannot be read properly. |

We use the red color to mark the poisonous word our method introduce.

Failed examples of MWPs generated by our *DMR* on the test set of Math23K are shown in Table 8. In the first case, our retrieval system may introduce a poisonous word "pencil" corresponding to the query (i.e., the input topic word "school" and "pens"). The introduced "pencil" confuses the logic of the generated MWPs. In the second case, the problem description of generated MWP meets the requirement. However, it asks a wrong question. The third case is interesting. The introduced phrase of the third case is exactly matched with the query. The introduced phrase "more than" between $N_0$ and $N_1$ corresponds to a part of the input equation(i.e., $N_0+N_1$). The introduced phrase "times as many as" near $N_2$ corresponds to a part of the input equation(i.e., $*N_2$). All the introduced phrases are matched with the query. However, since the order of the introduced phrase is wrong, the generated MWP is not as expected. This case shows that introducing matching retrieved results is not enough to generate ideal MWPs. The retrieved results still need to be well leveraged.

## 6 CONCLUSIONS

In this work, we introduce a memory retrieval module to the MWP generation framework and use the retrieved results to augment the input of the generation module. We observe that each MWP is composed of two parts (i.e., logical descriptions corresponding to the equation and scenario description corresponding to the topic words). To avoid the irrelevant information of the retrieved results, we propose a disentangled memory module that leverages the equation to retrieve the logical description memory and leverages the topic words to retrieve the scenario description memory. Experiments and case study show our superior performance and the effectiveness of each introduced module.

In the future, we can further explore on the basis of our approach. Considering that the mismatching retrieved results could damage the generation quality, we could try to improve the quality of the retrieved results. As shown in the third case of the Table 8, introducing matching retrieved results is not enough. Therefore, we could improve the generation module to fully utilize the retrieved results.

## ACKNOWLEDGMENTS

We thank Robert for explaining CMYK and color spaces.

## REFERENCES

[1] Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. 2019. Skeleton-to-response: Dialogue generation guided by retrieval memory. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL'19)*.

[2] Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, and Shuming Shi. 2019. Retrieval-guided dialogue response generation via a matching-to-generation framework. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19)*. Association for Computational Linguistics, 1866–1875. https://doi.org/10.18653/v1/D19-1195

[3] Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'21)*. 7307–7318.

[4] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1870–1879. https://doi.org/10.18653/v1/P17-1171

[5] Ricky T. Q. Chen, Xuechen Li, Roger B. Grosse, and David K. Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. *Adv. Neural Inf. Process. Syst.* 31 (2018).

[6] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. arXiv:1504.00325. Retrieved from https://arxiv.org/abs/1504.00325

[7] Paul Deane and Kathleen Sheehan. 2003. Automatic item generation via frame semantics: Natural language generation of math word problems.

[8] Cian Eastwood and Christopher K. I. Williams. 2018. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*.

[9] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1631–1640. https://doi.org/10.18653/v1/P16-1154

[10] Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Trans. Assoc. Comput. Ling.* 6 (2018), 437–450. https://doi.org/10.1162/tacl_a_00030

[11] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org, Article 368, 10 pages.

[12] Tatsunori B. Hashimoto, Kelvin Guu, Yonatan Oren, and Percy Liang. 2018. A retrieve-and-edit framework for predicting structured outputs. In *Proceedings of the Conference and Workshop on Neural Information Processing Systems (NeurIPS'18)*.

[13] Danqing Huang, Shuming Shi, Chin-Yew Lin, Jian Yin, and Wei-Ying Ma. 2016. How well do computers solve math word problems? large-scale dataset construction and evaluation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'16)*. 887–896.

[14] Shifeng Huang, Jiawei Wang, Jiao Xu, Da Cao, and Ming Yang. 2021. Recall and learn: A memory-augmented solver for math word problems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'21)*. Association for Computational Linguistics, 786–796. https://doi.org/10.18653/v1/2021.findings-emnlp.68

[15] Zhenya Huang, Xin Lin, Hao Wang, Qi Liu, Enhong Chen, Jianhui Ma, Yu Su, and Wei Tong. 2021. Disenqnet: Disentangled representation learning for educational questions. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 696–704.

[16] Jeffrey D. Karpicke. 2012. Retrieval-based learning: Active retrieval promotes meaningful learning. *Curr. Direct. Psychol. Sci.* 21 (2012), 157–163.

[17] Jeffrey D. Karpicke and Henry L. Roediger. 2008. The critical importance of retrieval for learning. *Science* 319 (2008), 966–968.

[18] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'20)*. Association for Computational Linguistics, Online, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[19] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

[20] Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1152–1157. https://doi.org/10.18653/v1/N16-1136

[21] Yihuai Lan, Lei Wang, Qiyuan Zhang, Yunshi Lan, Bing Tian Dai, Yan Wang, Dongxiang Zhang, and Ee-Peng Lim. 2021. MWPToolkit: An open-source framework for deep learning-based math word problem solvers. arXiv:2109.00799. Retrieved from https://arxiv.org/abs/2109.00799

[22] Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*. 228–231.

[23] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 6086–6096.
https://doi.org/10.18653/v1/P19-1612

[24] Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020. Pre-training via paraphrasing. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20)*. Curran Associates Inc., Red Hook, NY.

[25] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 9459–9474.

[26] Shucheng Li, Lingfei Wu, Shiwei Feng, Fangli Xu, Fengyuan Xu, and Sheng Zhong. 2020. Graph-to-tree neural networks for learning structured input-output translation with applications to semantic parsing and math word problem. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'20)*.

[27] Yanran Li, Ruixiang Zhang, Wenjie Li, and Ziqiang Cao. 2022. Hierarchical prediction and adversarial learning for conditional response generation. *IEEE Trans. Knowl. Data Eng.* 34, 1 (2022), 314–327. https://doi.org/10.1109/TKDE.2020.2977637

[28] Zhongli Li, Wenxuan Zhang, Chao Yan, Qingyu Zhou, Chao Li, Hongzhi Liu, and Yunbo Cao. 2022. Seeking patterns, not just memorizing procedures: Contrastive learning for solving math word problems. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'22)*.

[29] Zhenwen Liang, Jipeng Zhang, Lei Wang, Wei Qin, Yunshi Lan, Jie Shao, and Xiangliang Zhang. 2021. MWP-BERT: Numeracy-augmented pre-training for math word problem solving. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'21)*.

[30] Yongsub Lim, Minsoo Jung, and U. Kang. 2018. Memory-efficient and accurate sampling for counting local triangles in graph streams: From simple to multigraphs. *ACM Trans. Knowl. Discov. Data* 12, 1, Article 4 (Jan. 2018), 28 pages. https://doi.org/10.1145/3022186

[31] Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! numersense: Probing numerical commonsense knowledge of pre-trained language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'20)*. Association for Computational Linguistics, 6862–6868. https://doi.org/10.18653/v1/2020.emnlp-main.557

[32] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*. 74–81.

[33] Fenglin Liu, Xian Wu, Shen Ge, Xuancheng Ren, Wei Fan, Xu Sun, and Yuexian Zou. 2021. DiMBERT: Learning vision-language grounded representations with disentangled multimodal-attention. *ACM Transactions on Knowledge Discovery from Data* 16, 1, Article 1 (jul 2021), 19 pages. https://doi.org/10.1145/3447685

[34] Qianying Liu, Wenyv Guan, Sujian Li, and Daisuke Kawahara. 2019. Tree-structured decoding for solving math word problems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19)*. Association for Computational Linguistics, 2370–2379. https://doi.org/10.18653/v1/D19-1241

[35] Tianqiao Liu, Qiang Fang, Wenbiao Ding, Hang Li, Zhongqin Wu, and Zitao Liu. 2020. Mathematical word problem generation from commonsense knowledge graph and equations. arXiv:2010.06196. Retrieved from https://arxiv.org/abs/2010.06196

[36] Tianqiao Liu, Qian Fang, Wenbiao Ding, Zhongqin Wu, and Zitao Liu. 2021. Mathematical word problem generation from commonsense knowledge graph and equations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'21)*.

[37] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. 2019. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*. PMLR, 4114–4124.

[38] Yunshan Ma, Yujuan Ding, Xun Yang, Lizi Liao, Wai Keung Wong, and Tat-Seng Chua. 2020. *Knowledge Enhanced Neural Fashion Trend Forecasting*. Association for Computing Machinery, New York, NY, 82–90.

[39] Yunshan Ma, Xun Yang, Lizi Liao, Yixin Cao, and Tat-Seng Chua. 2019. Who, where, and what to wear? extracting fashion knowledge from social media. In *Proceedings of the 27th ACM International Conference on Multimedia (MM'19)*. Association for Computing Machinery, New York, NY, 257–265.

[40] Raheleh Makki, Eder Carvalho, Axel J. Soto, Stephen Brooks, Maria Cristina Ferreira De Oliveira, Evangelos Milios, and Rosane Minghim. 2018. ATR-Vis: Visual and interactive information retrieval for parliamentary discussions in twitter. *ACM Trans. Knowl. Discov. Data* 12, 1, Article 3 (Feb. 2018), 33 pages. https://doi.org/10.1145/3047010

[41] K. Nandhini and S. R. Balasundaram. 2011. Math word question generation for training the students with learning difficulties. In *Proceedings of the International Conference; Workshop on Emerging Trends in Technology*. 206–211.

[42] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'02)*. 311–318.

[43] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1532–1543.

[44] David Pfau, Irina Higgins, Alex Botev, and Sébastien Racanière. 2020. Disentangling by subspace diffusion. *Adv. Neural Inf. Process. Syst.* 33 (2020), 17403–17415.

[45] Oleksandr Polozov, Eleanor O'Rourke, Adam M. Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popovic. 2015. Personalized mathematical word problem generation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'15)*.

[46] Wei Qin, Zetong Chen, Lei Wang, Yunshi Lan, Wei Ren, and Richang Hong. 2023. Read, diagnose and chat: Towards explainable and interactive LLMs-augmented depression detection in social media. arXiv:2305.05138. Retrieved from https://arxiv.org/abs/2305.05138

[47] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1 (2019), 9.

[48] Doug Rohrer and Harold Pashler. 2010. Recent research on human learning challenges conventional instructional strategies. *Educ. Res.* 39 (2010), 406–412.

[49] Martha Roseberry, Bartosz Krawczyk, and Alberto Cano. 2019. Multi-label punitive KNN with self-adjusting memory for drifting data streams. *Proceedings of Machine Learning Research* 13, 6, Article 60 (Nov. 2019), 31 pages. https://doi.org/10.1145/3363573

[50] Abigail See, Peter Liu, and Christopher Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'17)*.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Conference and Workshop on Neural Information Processing Systems (NeurIPS'17)*.

[52] Lieven Verschaffel, Stanislaw Schukajlow, Jon Star, and Wim Van Dooren. 2020. Word problems in mathematics education: A survey. *ZDM* 52 (2020), 1–16.

[53] Candace A. Walkington. 2013. Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes. *J. Educ. Psychol.* 105 (2013), 932.

[54] Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[55] Lei Wang, Dongxiang Zhang, Jipeng Zhang, Xing Xu, Lianli Gao, Bing Tian Dai, and Heng Tao Shen. 2019. Template-based math word problem solvers with recursive neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 7144–7151. https://doi.org/10.1609/aaai.v33i01.33017144

[56] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2724–2743. https://doi.org/10.1109/TKDE.2017.2754499

[57] Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 845–854. https://doi.org/10.18653/v1/D17-1088

[58] Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'17)*. 845–854.

[59] Zichao Wang, Andrew Lan, and Richard Baraniuk. 2021. Math word problem generation with mathematical consistency and problem context constraints. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'21)*.

[60] Jason Weston, Emily Dinan, and Alexander H. Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'18)*.

[61] Sandra Williams. 2011. Generating mathematical word problems. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'11)*.

[62] Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. 2022. Graph neural networks for natural language processing: A survey. *Found. Trends Mach. Learn.* (2022).

[63] Yu Wu, Furu Wei, Shaohan Huang, Yunli Wang, Zhoujun Li, and Ming Zhou. 2019. Response generation by context-aware prototype editing. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'19)*.

[64] Han Xiao, Yidong Chen, and Xiaodong Shi. 2021. Knowledge graph embedding based on multi-view clustering framework. *IEEE Trans. Knowl. Data Eng.* 33, 2 (2021), 585–596. https://doi.org/10.1109/TKDE.2019.2931548

[65] Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*. International Joint Conferences on Artificial Intelligence Organization, 5299–5305. https://doi.org/10.24963/ijcai.2019/736

[66] Xun Yang, Fuli Feng, Wei Ji, Meng Wang, and Tat-Seng Chua. 2021. Deconfounded video moment retrieval with causal intervention. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1–10.

[67] Jipeng Zhang, Roy Ka-Wei Lee, Ee-Peng Lim, Wei Qin, Lei Wang, Jie Shao, and Qianru Sun. 2020. Teacher-student networks with multiple decoders for solving math word problem. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'20)*.

[68] Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales?. In *Findings of the Association for Computational Linguistics: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'20)*. Association for Computational Linguistics, 4889–4896. https://doi.org/10.18653/v1/2020.findings-emnlp.439

[69] Qingyu Zhou and Danqing Huang. 2019. Towards generating math word problems from equations and topics. In *Proceedings of the 12th International Conference on Natural Language Generation (INLG'19)*.