Disentangled Memory Retrieval Towards Math Word Problem Generation

Anonymous ACL submission

Abstract

The task of math word problem (MWP) generation, which generates a MWP given an equation and relevant topic words, has increasingly attracted researchers' attention. In this work, we propose a seq2seq model with a disentangled memory retrieval module to better take advantage of the logical description and scenario description within a MWP and more relevant training data to improve the generation quality. We first disentangle the training MWPs into logical descriptions and scenario description and then record them in respective memory modules. Later, we use the given equation and topic words as queries to retrieve the most relevant logical descriptions and scenario description from the corresponding memory modules respectively. The retrieved results are then used to complement the process of the MWP generation. Extensive experiments verify the superior performance and effectiveness of our method. The code is available on https://github.com/mwp-g/MWPG-DMR.

1 Introduction

Math word problems play 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 (Walkington, 2013; Wang et al., 2018; Zhang et al., 2020; Verschaffel et al., 2020; Wang et al., 2021). As shown in Table 1, an MWP consists of a question and a corresponding equation, and the question is composed of the *logical description* marked by the orange color and the *scenario 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 (Verschaffel et al., 2020). Many studies (Karpicke and Roediger, 2008; Karpicke, 2012; Rohrer and Pashler, 2010) have showed that high-quality MWPs could lead to better engagement and improve the learning outcomes. However, manually designing MWPs by experts costs a lot and the qualities of the generated MWPs heavily rely on the experts.

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
Equation:	$N_0 * N_1 + N_2 + N_0$ ($23 * 2 + 6 + 23$)

In this paper, we focus on the problem of automated math word problem generation, 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 (Deane and Sheehan, 2003; Polozov et al., 2015; Williams, 2011; Nandhini and Balasundaram, 2011). However, the language quality and diversity of MWPs generated by text templates are not as expected. Recently, some models (Huang et al., 2016; Liu et al., 2021; Wang et al., 2021) based on deep neural networks have bought significant improvement in generating MWPs. However, since the generation process of those methods only conditions on the given topic words and equation, the scenario description lacks richness and the logical description lacks equation-consistency. As shown in Figure 1(a), the generation of *seq2seq* lacks some keywords of scenario description(such as *farm*) and the logical description is inconsistent with the input equation.

To generate more rich scenario description and more consistent logical description with equation , we introduce a memory-retrieved module, which takes full advantage of the training MWPs, into the framework. Memory-retrieved module has been shown to facilitate a number of text generation tasks such as dialogue generation (Weston et al., 2018; Cai et al., 2019; Wu et al., 2019), machine translation (Cai et al., 2021), and code generation (Hashimoto et al., 2018). To this goal, we record all the training MWPs into the memory in

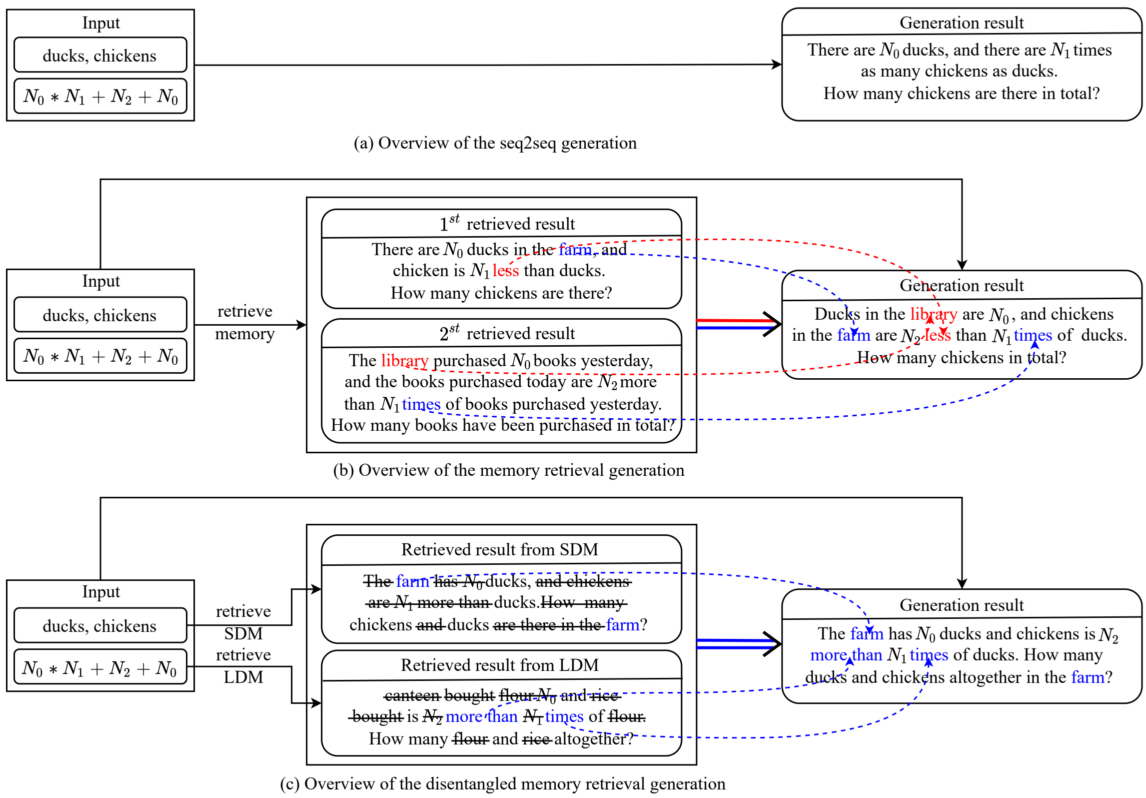

Figure 1: Illustration about different genneration models

advance. During inference, we utilize the most related MWPs retrieved from the memory module to complement the generation condition, i.e., the topic words and the equation. As shown in Figure 1(b), the 1^{st} retrieved result from the single memory module introduces a new word of scenario description *farm* corresponding to *ducks* and *chickens*, improving the richness of the scenario description. The 2^{nd} retrieved result introduces a new word of logical description *times* corresponding to the multiplication sign, improving the equation-consistency. MWPs are composed of logical description and scenario description, which are entangled in a MWP. Therefore, the retrieved MWPs with good scenario description does not necessarily have good logical description, and vice versa. For example, as shown in Figure 1(b), the scenario description *library* from the 2^{nd} retrieved result is irrelevant to the input topic words(i.e., *ducks*, *chickens*). The logical description from the 1^{st} retrieve result, i.e., *less*, and the input equation $N_0 * N_1 + N_2 + N_0$ mutually contradict. Apparently, introducing those irrelevant information (i.e., *library* and *less*) into the generation module will damage the quality of the generated MWPs.

To alleviate this issue, we propose a disentangled memory retrieval framework. As shown in Figure 1, we disentangle the training MWPs into the scenario description corresponding to the topic words and the logical description corresponding to the equations. Then, we utilize the disentangled MWPs to build the scenario description memory (SDM) and the logical description memory (LDM) individually. During inference, we obtain the most related scenario description by leveraging the topic words to retrieve the SDM and the most related logical description by leveraging the equation to retrieve the LDM. Both the retrieved scenario description and logical description will complement the generation condition. As shown in Figure 1(c), the input topic words *ducks* and *chicken* retrieve *farm* from SDM, improving the richness of scenario description. The equation $N_0 * N_1 + N_2 + N_0$ retrieves *more than ... times* from the LDM, improving the equation-consistency of the generated MWP. We name the framework as Math Word Problem Generation via Disentangled Memory Retrieval, MWPG-DMR. The contributions are as follows:

- To the best of our knowledge, we are the first work that introduces the memory module into the math word problem generation;
- Inspired by the observation that MWPs are composed of logical descriptions corresponding to equation and scenario description corresponding to topic words, we propose a disentangled memory retrieval framework for generating math word problems;

- The MWPG-DMR significantly outperforms 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 (Deane and Sheehan, 2003; Polozov et al., 2015; Williams, 2011; Nandhini and Balasundaram, 2011). Recently, some models based on deep neural networks have bought significant improvement in generating MWPs. MCPCC (Huang et al., 2016), based on a standard encoder-decoder architecture, forces the entities in the generated MWP to correspond to the variables in the input equation . The works in (Liu et al., 2021) fuses information from equations and commonsense knowledge to facilitate the generation. And the work in (Wang et al., 2021), 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, since the generation process of those methods only conditions on the given topic words and equation, the scenario description lacks richness and the logical description lacks equation-consistency.

Text generation with retrieval. Memory-retrieved module has been shown to facilitate a number of text generation tasks such as dialogue generation (Weston et al., 2018; Cai et al., 2019; Wu et al., 2019), machine translation (Cai et al., 2021), and code generation (Hashimoto et al., 2018; Huang et al., 2021a). It is obvious that the retrieval algorithm can solve a particular task by constructing a knowledge base, which is suitable for the generator.

Disentanglement. There are various definitions for disentanglement (Schmidhuber, 1992; Eastwood and Williams, 2018; Chen et al., 2018), but a common goal is a latent space that consists of linear subspaces, each of which controls one factor of variation. So disentanglement is usually used when an entity has multiple parts. Many works in different fields such as representation learning (Huang et al., 2021b; Pfau et al., 2020; Locatello et al., 2019), image generation (Karras et al., 2019; Pidhorskyi et al., 2020), and moment retrieval (Yang et al., 2021) had adopted disentanglement to make they data be better represented, so that the model learns what it wants more accurately.

3 Problem Setup and Notations

Following (Wang et al., 2021), we formulate MWP generation as a task of multi-view (topic words and an equation) conditional text generation. Then, we describe the MWP generation process as:

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

where the datasets are denoted as $\mathcal{D} = \{M_i, x_i^{eq}, x_i^{tw}\}_{i=1}^N$. x_i^{eq}, x_i^{tw}, denoting the equation and topic words respectively, are the generation conditions. $M_i = \{m_1, ..., m_T\}$, as the generation target, represents the MWP as a sequence of T tokens. p_Θ denotes the MWP generation model parameterized by a set of parameters. The generation model p_Θ condition on topic words x_i^{tw} and equation x_i^{eq} and generate the MWP $\hat{M}_i = \{\hat{m}_1, ..., \hat{m_{T'}}\}$. The generated MWP $\hat{M}_i$ is expected to be same with the generation target M_i and consistent with the input equation x_i^{eq}. We will discuss the detailed evaluation metric in section 5.

4 Proposed Approach

4.1 Overview of the proposed approach

We will elaborate the proposed approach in the next 4 subsections.

Pre-processing stage In this stage, we disentangle all the 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 then build the logical description memory(LDM) and scenario description memory(SDM). We will elaborate the details of the preprocess in section 4.2.

The disentangled retrieval module In this module, we use the topic words x_i^{tw} and equation x_i^{eq} to retrieve SDM and LDM, built by disentangling the training MWPs in the pre-processing stage, respectively. The disentangled retrieval module consists of the topic-words-based retrieval module and the equation-based retrieval module. In specific, given the input (x_i^{tw}, x_i^{eq}), the topic-words-retrieval module selects a number of possibly helpful scenario description $\{M_j^{sd}\}_{j=1}^{N_{sd}}$ from SDM, according to a relevant function $f_{sd}(x_i^{tw}, M_j^{sd})$. Similarly, the equation-based-retrieval module selects a number of possibly helpful logical descriptions $\{M_j^{ld}\}_{j=1}^{N_{ld}}$ from LDM, according to a relevant function $f_{ld}(x_i^{eq}, M_j^{ld})$. We will elaborate the disentangled retrieval module in the section 4.3.

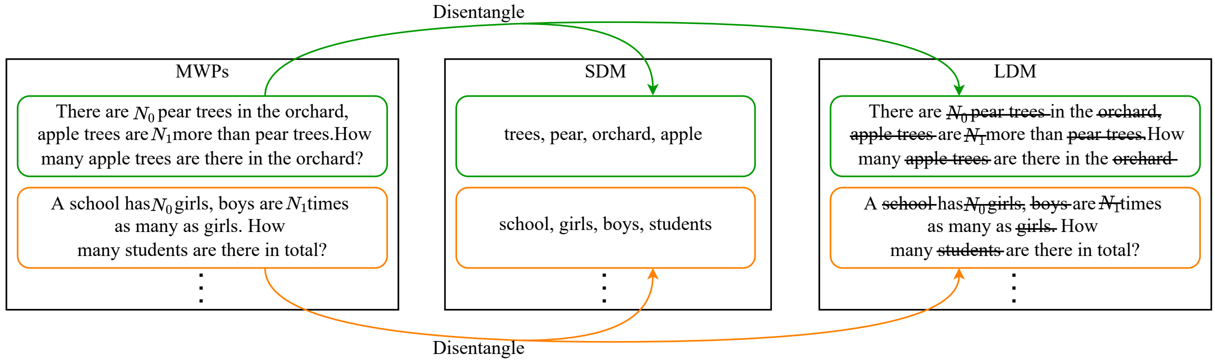

Figure 2: Disentangle the training MWPs and build SDM and LDM

The generation module The generation module conditions on both the retrieved results $(\{M_j^{sd}\}_{j=1}^{N_{sd}}, \{M_j^{ld}\}_{j=1}^{N_{ld}})$ and the original inputs (x_i^{tw}, x_i^{eq}) to generate the output $\hat{M}_i$. The generation module can be described as: $p(\hat{M}_i|x_i^{tw}, x_i^{eq}, M_1^{sd}, ..., M_{N_{sd}}^{sd}, M_1^{ld}, ..., M_{N_{ld}}^{ld})$. In section 4.4, we will elaborate the generation module.

The training process In section 4.5, we will elaborate the details of the training process and the pretraining process.

4.2 Pre-processing

In the pre-processing stage, 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 logical description memory(LDM) and scenario description memory(SDM).

Disentangle the training MWPs Following (Hosseini et al., 2014) , we assume the scenario description is mainly described by nouns and the logical description is described by the other words including verbs, adverbs, prepositions and so on. Therefore, we use the TF-IDF to identify the two part in the MWP. And, as shown in Figure 2, we extract the nouns in the MWP M_i as its scenario description M_i^{sd}. The others words except numbers and the mask token replacing the nouns are regarded as its logical description M_i^{ld}. Unlike the nouns in the scenario description, the position of the words in the logical description may influence the semantic. Therefore, we preserve the position of the words in the logical description.

Build Memory Further, as shown in Figure 2, we record all logical description $\{M_i^{ld}\}_{i=1}^{N}$ and scenario description $\{M_i^{sd}\}_{i=1}^{N}$ into logical description memory(LDM) and scenario description memory(SDM) respectively.

4.3 Disentangled Retrieval Module

Compared with conventional retrieval module that used the joint query(topic words and equation) to retrieve all the training MWPs, our disentangled retrieval module use the topic words x_i^{tw} and the equation x_i^{eq} to retrieve the SDM and LDM, which are the disentangled results from all the training MWPs, respectively. In specific, the disentangled retrieval module consists of a topic-words-based retrieval module and an equation-based retrieval module.

Topic-words-based Retrieval Module Given the input topic words x_i^{tw} and the scenario description memory (SDM), the topic-words-based retrieval module retrieves the top N_{sd} relevant scenario descriptions $\{M_j^{sd}\}_{j=1}^{N_{sd}}$, according to the relevance score $f_{tw}(x_i^{tw}, M_j^{sd})$. We define the relevance score $f_{tw}(x_i^{tw}, M_j^{sd})$ between the input topic words x_i^{tw} and each candidate scenario description M_j^{sd} as the inner product of their representations:

$$f_{tw}(x_i^{tw}, M_j^{cn}) = ENC_{tw}(x_i^{tw})^T ENC_{sd}(M_j^{sd}) \tag{2}$$

where ENC_{tw} and ENC_{sd} are the input topic words encoder and the scenario description encoder that encode x_i^{tw} and M_j^{sd} to d-dimensional vectors respectively.

$$ENC_{tw}(x_i^{tw}) = normalize(W_{tw} Tr_{tw}(x_i^{tw})) \tag{3}$$

$$ENC_{sd}(M_j^{sd}) = normalize(W_{sd} Tr_{cn}(M_j^{sd})) \tag{4}$$

where Tr_{tw} is the Transformer(Vaswani et al., 2017) encoder of the input topic words x_i^{tw}. Tr_{sd} is the Transformer encoder of the scenario description M_j^{cn}. W_{tw} and W_{sd} are the matrices of the linear projections, which reduce the dimension of

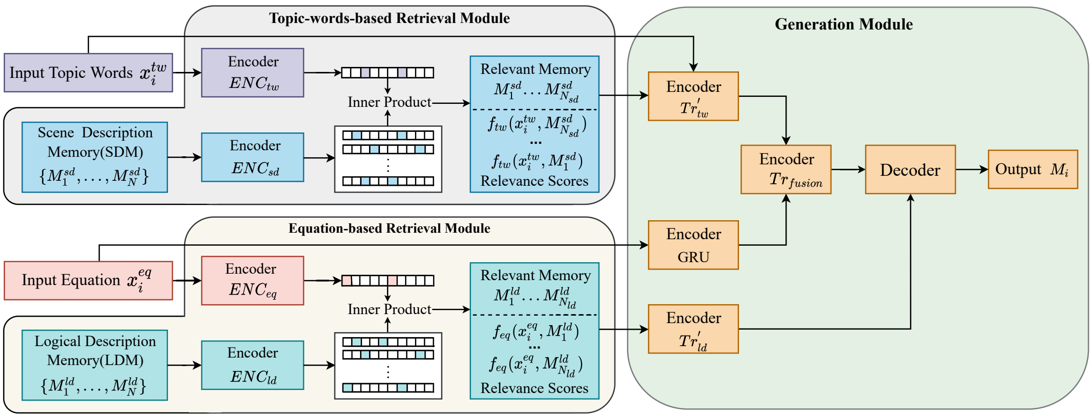

Figure 3: The framework of our DMR

the representations. $normalize()$ could normalize any vector to a unit vector and is used to regulate the range of the relevance score.

Equation-based Retrieval Module Given the input equation x_i^{eq} and the logical description memory(LDM), the equation-based retrieval module retrieves the top N_{ld} relevant $\{M_j^{ld}\}_{j=1}^{N_{ld}}$, according to the relevance score $f_{eq}(x_i^{eq}, M_j^{ld})$, which is defined as follows:

$$f_{eq}(x_i^{eq}, M_j^{ld}) = ENC_{eq}(x_i^{eq})^T ENC_{ld}(M_j^{ld}) \tag{5}$$

$$ENC_{eq}(x_i^{eq}) = normalize(W_{eq} GRU_{eq}(x_i^{eq})) \tag{6}$$

$$ENC_{ld}(M_j^{ld}) = normalize(W_{ld} Tr_{ld}(M_j^{ld})) \tag{7}$$

where the function of GRU_{eq}, Tr_{ld}, W_{eq}, and W_{ld} are similar to Tr_{tw}, Tr_{sd}, W_{tw}, and W_{sd} mentioned in Eq.2-4, respectively. In Eq.6, we employ GRU_{eq} rather than Transformer to encode the equation x_i^{eq}, since GRU pays more attention to the order of the sequence. And we actually use the equation in the form of postfix expression in which the sequence order can represent the calculation order, so useing GRU, the whole model can learn more about the meaning of mathematical formulas.

4.4 Generation Module

Conditioned on both the original input (x_i^{tw}, x_i^{eq}) and the retrieved results $(\{M_j^{sd}\}_{j=1}^{N_{sd}}, \{M_j^{ld}\}_{j=1}^{N_{ld}})$ 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 $p(\hat{M}_i | x_i^{tw}, x_i^{eq}, M_1^{sd}, ..., M_{N_{sd}}^{sd}, M_1^{ld}, ..., M_{N_{ld}}^{ld})$. Since the retrieved scenario description $\{M_j^{sd}\}_{j=1}^{N_{sd}}$ 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 $\{M_j^{ld}\}_{j=1}^{N_{ld}}$ contains the structure information, we copy the retrieved logical description into generation via the cross attention mechanism (See et al., 2017). 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 $(\{M_j^{cn}\}_{j=1}^{N_{cn}}, \{M_j^{ld}\}_{j=1}^{N_{ld}})$ into representations:

$$v_i^{tw} = Tr'_{tw}(x_i^{tw}, M_1^{sd}, ..., M_{N_{sd}}^{sd}) \tag{8}$$

$$v_i^{eq} = GRU(x_i^{eq}) \tag{9}$$

$$v_i^{fs} = Tr_{fusion}(v_i^{tw}, v_i^{eq}) \tag{10}$$

$$V_i^{ld} = \{Tr'_{ld}(M_j^{ld})\}_{j=1}^{N_{ld}} \tag{11}$$

In eq.8, the Transformer Tr'_{tw} encodes the input topic words x_i^{tw} and retrieved scenario descriptions $\{M_j^{cn}\}_{j=1}^{N_{cn}}$ into the representation v_i^{tw}. In eq.9, the GRU encodes the input equation x_i^{eq} into the representation v_i^{eq}. In eq.10, the Transformer Tr_{fusion} fuses v_i^{tw} and v_i^{eq} into v_i^{fs}. In eq.11, the logical description Transformer encoder Tr'_{ld} encodes each the retrieved logical description $\{M_j^{ld}\}_{j=1}^{N_{ld}}$ individually, resulting in a set of representations V^{ld}.

The decoder can be regarded as a probabilistic model $p(M_i | v_i^{fs}, V_i^{ld})$. Fed with the presentations v_i^{fs} and V_i^{ld}, the decoder generates an output sequence M_i in an auto-regressive fashion. At each time step 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, i.e., $P_v = softmax(W_v h_t + b_v)$. In addition, we also compute a cross attention over the representation of all retrieved logical description $V_i^{ld} = \{Tr'_{ld}(M_j^{ld})\}_{j=1}^{N_{ld}}$:

$$\alpha_{ij} = \frac{exp(h_t^T W_m V_{i,j}^{ld})}{\sum_{i=1}^{M}\sum_{k=1}^{L_i} exp(h_t^T W_m v_{i,k})} \tag{12}$$

$$c_t = W_c \sum_{i=1}^{M}\sum_{j=1}^{L_i} \alpha_{ij} V_{i,j}^{ld} \tag{13}$$

where V_{ij}^{ld} is the j-th token in the i-th logical description. L_i denote the length of the i-th retrieved logical description M_i. α_{ij} is the attention score of V_{ij}^{ld}, c_t is a weighted combination of memory embeddings, and W_m and W_c are trainable matrices. The next-token probabilities are computed as:

$$p(m_t|\cdot) = (1-\lambda_t)P_v(m_t) + \lambda_t \sum_{j=1}^{M}\sum_{j=1}^{L_i} \alpha_{ij} \mathbb{1}_{V_{ij}^{ld}=m_t} \tag{14}$$

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

4.5 Training

We optimize the parameters Θ of the model using stochastic gradient descent(SGD) on the negative log-likelihood loss function $-\log p(M_i|x_i^{tw}, x_i^{eq}, M_1^{sd}, ..., M_{N_{sd}}^{sd}, M_1^{ld}, ..., M_{N_{ld}}^{ld})$ 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 $\{x_i^{tw}, M_i^{sd}\}_{i=1}^{N}$ from training set and SDM at each training step. Let $X_{tw} \in R^{B\times b}$ and $P_{sd} \in R^{B\times b}$ be the representation of the topic words and scenario description through ENC_{tw} and ENC_{sd} respectively. $S = X_{tw}P_{sd}^T$ 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 slot. Any $(X_{tw,i}, P_{sd,j})$ 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:

$$\mathcal{L}_{tw}^{(i)} = \frac{-exp(S_{ii})}{exp(S_{ii}) + \sum_{j\neq i} exp(S_{ij})} \tag{15}$$

Pre-training for equation-based retrieval module We sample all equation and logical-description pairs $\{x_i^{eq}, M_i^{ld}\}_{i=1}^{N}\}$ from the training set and LDM at each training step. Let $X_{eq} \in R^{B\times b}$ and $P_{ld} \in R^{B\times b}$ be the representation of the equation and logical description through ENC_{eq} and ENC_{ld} respectively. Similar to S in Equ. 15, $U = X_{eq}P_{ld}^T$ 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:

$$\mathcal{L}_{eq}^{(i)} = \frac{-exp(U_{ii})}{exp(U_{ii}) + \sum_{j\neq i} exp(U_{ij})} \tag{16}$$

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 (Wang et al., 2017), MAWPS (Koncel-Kedziorski et al., 2016) and Dolphine18K (Huang et al., 2016). Following the splitting strategy of (Lan et al., 2021), we split each dataset into trainset, validation set and test set. The summary statistics of datasets are shown in Table 2.

Table 2: Summary statistics of datasets

Dataset	#trainset	#valset	#testset	total
Math23K	16781	2083	2111	20975
Dolphin18K	7593	847	2110	10550
MAWPS	1865	241	241	2347

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 (Zhou and Huang, 2019), we concatenate multiple equations as a single equation.

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-pre	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
DMR(ours)	**0.634**	**0.545**	**0.758**	**0.605**	**0.388**	**0.372**	**0.627**	**0.545**

Table 4: Experiment results on Dolphin18K

Models	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
DMR (ours)	0.228	0.339	0.478

Table 5: Ablation study

Models	BLEU-4	METEOR	ROUGLE-L	ACC-eq
seq2seq(ours)	0.310	0.329	0.526	0.490
seq2seq(ours) w/ memory	0.330	0.333	0.545	0.506
DMR(ours)	0.388	0.372	0.627	0.545

Baselines In Table 3, *seq2seq-rnn*, based on the LSTMs with attention (Zhou and Huang, 2019; Liu et al., 2020), 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 (Pennington et al., 2014) instead of random embeddings at initialization and *seq2seq-tf* is based on Transformers (Vaswani et al., 2017) rather than RNN. We also compare our approach to vanilla GPT-2 (Radford et al., 2019), either finetuned 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 (Wang et al., 2021). In Table 4, *MaGNET* (Zhou and Huang, 2019), 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 (Wang et al., 2021; Zhou and Huang, 2019), experiments are only performed on part of those three datasets. Therefore, our method is compared with different baselines on different datatsets.

Ablation Study Baselines 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 with 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.

Metrics We leverage the following three commonly used evaluation metrics: BLEU-4 (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007) and ROUGE (Lin, 2004) to measure the language quality. We implement those three metrics using the package provided by (Chen et al., 2015). 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, our *DMR* achieves better language quality and equation consistency than both seq2seq-based methods and GPT-based methods on Math23K and MAWPS. However, the metric ACC-eq of all the methods is not good enough. ACC-eq equals 60.5% and 54.5% on the MAWPS and Math23K respectively. In other words, at least 39.5% and 45.5% of the generated MWPs are unusable, since their logical description is inconsistent with their equations.

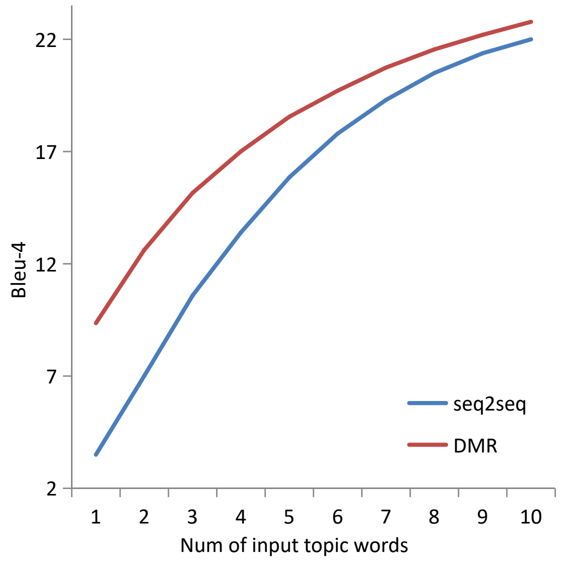

Figure 4: Experiments with different max numbers of topic words words as input.

Table 4 shows our *DMR* outperforms the best baselines on Dolphin18K. The quantitative results on Dolphin18K is lower than results on MAWPS and Math23k, since generating MWPs on a multiple-equation MWP datset is much more difficult. The metric ACC-eq of multiple-equation MWP dataset is difficult to calculate and thus ACC-eq is not used on Dolphin18K.

Ablation Study We can find that *seq2seq(ours) w/ memory* performs slightly better than *seq2seq(ours)*. This shows that the retrieved results from the single memory improve the language quality of the generated MWPs. However, the improvement is limited. According to the case study in Figure 1, we can speculate that this is because not all information of the retrieved results is beneficial. Our *DMR* achieves much better performance than *seq2seq(ours) w/ memory*. Therefore, we can conclude that the disentangled memory retrieval(DMR) is better than the single memory retrieval.

Number of the input topic words To verify the claim in section 1 that our method could improve the richness of the scenario description, we perform experiments with different number of input topic words on Dolphin18K. As shown in Figure 4, the fewer topic words we input, the greater the gap between the our *DMR* and the **seq2seq**. A small number of topic words in the training examples means that they do not fully summarize the scenarios of the MWPs. However, our *DMR* still achieve higher BLEU value by generating MWPs as similar as possible to the ground-truth MWPs. Also based on the case study, we can conclude that our *DMR* could improve the richness of the scenario description by augmenting the topic words with retrieved scenario description.

Table 6: Human evaluation results

Models	Equation Relevance	Topic Words Relevance	Language Fluency
Seg2Seq-rnn	1.71	2.34	2.19
Seq2Seq-tf	2.17	2.57	2.55
GPT-pre	2.24	2.71	2.60
MCPCC	2.42	2.80	2.64
DMR	**2.54**	**2.88**	**2.76**

5.2 Qualitative Results

Case Study Cases in Figure 1 is real cases from the generation results of test set. From Figure 1(a), the scenario description 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 single memory of *seq2seq w/ memory* 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 memory. More case study is presented in section Appendix.

Human Evaluation In addition, because automatic evaluation metrics are not always consistent with human judgments on the correctness of a math word problem,we conducted human evaluation on our model compared with several baselines mentioned above. We consider three metrics:

- Equation Relevance: a problem is relevant to the given equation;
- Topic Word Relevance: a problem is relevant to all given topic words;
- Language Fluency: 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 1-3 scale (3 for 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.

6 Conclusions

In this work, we observe that each MWP is composed of two parts: logical descriptions corresponding to the equation and context narratives corresponding to the topic words. We design a disentangled memory module which leverages the equation to retrieve the logical description memory and leverages the topic words to retrieve the context narrative memory. Experiments show our superior performance and the effectiveness of each introduced module.

Acknowledgements

This document has been adapted by Steven Bethard, Ryan Cotterell and Rui Yan from the instructions for earlier ACL and NAACL proceedings, including those for ACL 2019 by Douwe Kiela and Ivan Vulić, NAACL 2019 by Stephanie Lukin and Alla Roskovskaya, ACL 2018 by Shay Cohen, Kevin Gimpel, and Wei Lu, NAACL 2018 by Margaret Mitchell and Stephanie Lukin, BibTEX suggestions for (NA)ACL 2017/2018 from Jason Eisner, ACL 2017 by Dan Gildea and Min-Yen Kan, NAACL 2017 by Margaret Mitchell, ACL 2012 by Maggie Li and Michael White, ACL 2010 by Jing-Shin Chang and Philipp Koehn, ACL 2008 by Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, ACL 2005 by Hwee Tou Ng and Kemal Oflazer, ACL 2002 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence* and the *Conference on Computer Vision and Pattern Recognition*.

References

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 *NAACL)*.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In *ACL)*, pages 7307–7318.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31.

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 preprint arXiv:1504.00325*.

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

Cian Eastwood and Christopher KI Williams. 2018. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*.

Tatsunori B. Hashimoto, Kelvin Guu, Yonatan Oren, and Percy Liang. 2018. A retrieve-and-edit framework for predicting structured outputs. In *NeurIPS*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *EMNLP*, pages 523–533.

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 *ACL*, pages 887–896.

Shifeng Huang, Jiawei Wang, Jiao Xu, Da Cao, and Ming Yang. 2021a. Recall and learn: A memory-augmented solver for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 786–796, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Jeffrey D Karpicke. 2012. Retrieval-based learning: Active retrieval promotes meaningful learning. *Current Directions in Psychological Science*, 21:157–163.

Jeffrey D Karpicke and Henry L Roediger. 2008. The critical importance of retrieval for learning. *science*, 319:966–968.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of* *the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

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 preprint arXiv:2109.00799*.

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 second workshop on statistical machine translation*, pages 228–231.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tianqiao Liu, Qian Fang, Wenbiao Ding, Zhongqin Wu, and Zitao Liu. 2021. Mathematical word problem generation from commonsense knowledge graph and equations. In *EMNLP*.

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 preprint arXiv:2010.06196*.

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*, pages 4114–4124. PMLR.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

David Pfau, Irina Higgins, Alex Botev, and Sébastien Racanière. 2020. Disentangling by subspace diffusion. *Advances in Neural Information Processing Systems*, 33:17403–17415.

Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. 2020. Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14104–14113.

Oleksandr Polozov, Eleanor O'Rourke, Adam M. Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popovic. 2015. Personalized mathematical word problem generation. In *IJCAI*.

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

Doug Rohrer and Harold Pashler. 2010. Recent research on human learning challenges conventional instructional strategies. *Educational Researcher*, 39:406–412.

Jürgen Schmidhuber. 1992. Learning factorial codes by predictability minimization. *Neural computation*, 4(6):863–879.

Abigail See, Peter Liu, and Christopher Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS*, 30.

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

Candace A Walkington. 2013. Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes. *Journal of Educational Psychology*, 105:932.

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*, volume 32.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *EMNLP*, pages 845–854.

Zichao Wang, Andrew Lan, and Richard Baraniuk. 2021. Math word problem generation with mathematical consistency and problem context constraints. In *EMNLP*.

Jason Weston, Emily Dinan, and Alexander H. Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. In *EMNLP*.

Sandra Williams. 2011. Generating mathematical word problems. In *AAAI*.

Yu Wu, Furu Wei, Shaohan Huang, Yunli Wang, Zhoujun Li, and Ming Zhou. 2019. Response generation by context-aware prototype editing. *AAAI*, 33.

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*, pages 1–10.

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. IJCAI.

Qingyu Zhou and Danqing Huang. 2019. Towards generating math word problems from equations and topics. In *ICNLG*.

A Training details

Table 7 provides the configurations for our method and all baselines. Experiments performed on all datasets use the same configuration. Each model are trained on two NVIDIA RTX 3090 GPUs.

B Case study

Table 8 presents additional examples of MWPs generated by our method. All the generated examples are taken from the Math23K dataset. These examples are consistent with the qualitative results in Figure 1.

Table 7: Model configurations.

architecture	#layers	input size	layer size	#params	optimizer	learning rate	batch size	training epoch/steps
seq2seq-rnn	2	300	512	11M	adagrad	0.15	64	{5000, 15000∗}
seq2sea-tf	6	512	512	52M	Adam	2	4096	{5000, 15000∗}
GPT	36	1280	1280	774M	Adam	5e-5	8	{15000, 40000∗}
MCPCC	36	1280	1280	774M	Adam	5e-5	16	{1000, 3000∗}
DMR	11	512	512	59.39M	Adam	1.4e-6	512	{8000, 15000∗}

Table 8: Additional examples of MWPs generated by our approach

Equation	N_0/N_1
Topic words	village, canal
Ground truth	The village needs to dig a N_0 kilometers canal, digging N_1 kilometers every day. How many days can it be dug?
Gen.MWP	The village needs to dig a N_0 kilometers canal. It planned to dig N_1 kilometers every day. How many days will it take to complete the canal?
Equation	$N_0 + N_1$
Topic words	mother, vegetables
Ground truth	My mother spent N_0 yuan to buy vegetables, and there is still N_1 yuan left. How much money did my mother bring?
Gen.MWP	My mother went to the street to buy vegetables, spent N_0 yuan, and there was N_1 yuan left. How much money did mom bring?
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	school, storybooks
Ground truth	The school plans to distribute storybooks to N_0 classes, N_1 for each class,and N_2 for spare. How many storybooks should the school prepare?
Gen.MWP	The school bought N_0 storybooks and bought comics N_2 more than N_1 times the number of storybooks. How many comics did the school buy?