Modeling Intra-Relation in Math Word Problems with Different Functional Multi-Head Attentions

Jierui Li[1], **Lei Wang**[1]*, **Jipeng Zhang**[1], **Yan Wang**[2], **Bing Tian Dai**[3], **Dongxiang Zhang**[45]

[1]Center for Future Media and School of Computer Science & Engineering, UESTC, [2]Tencent AI Lab

[3]School of Information Systems, Singapore Management University, [4]Afanti Research, [5]Zhejiang University

{lijierui, zhangjipeng20}@std.uestc.edu.cn, demolei@outlook.com

bradenwang@tencent.com, btdai@smu.edu.sg, zhangdongxiang37@gmail.com

Abstract

Several deep learning models have been proposed for solving math word problems (MWPs) automatically. Although these models have the ability to capture features without manual efforts, their approaches to capturing features are not specifically designed for MWPs. To utilize the merits of deep learning models with simultaneous consideration of MWPs' specific features, we propose a group attention mechanism to extract global features, quantity-related features, quantity-pair features and question-related features in MWPs respectively. The experimental results show that the proposed approach performs significantly better than previous state-of-the-art methods, and boost performance from 66.9% to 69.5% on Math23K with training-test split, from 65.8% to 66.9% on Math23K with 5-fold cross-validation and from 69.2% to 76.1% on MAWPS.

1 Introduction

Computer systems, dating back to 1960s, have been developing to automatically solve math word problems (MWPs) (Feigenbaum and Feldman, 1963; Bobrow, 1964). As illustrated in Table 1, when solving this problem, machines are asked to infer "how many shelves would Tom fill up" based on the textual problem description. It requires systems having the ability to map the natural language text into the machine-understandable form, reason in terms of sets of numbers or unknown variables, and then derive the numeric answer.

In recent years, a growing number of deep learning models for MWPs (Wang et al., 2017; Ling et al., 2017; Wang et al., 2018b,a; Huang et al., 2018a,b; Wang et al., 2019) have drawn inspiration from advances in machine translation.

Problem: For a birthday party Tom bought 4 regular sodas and 52 diet sodas. If his fridge would only hold 7 on each shelf, how many shelves would he fill up?
Equation: $x = (4.0 + 52.0)/7.0$ **Solution**: 8

Table 1: A math word problem.

The core idea is to leverage the immense capacity of neural networks to strengthen the process of equation generating. Compared to statistical machine learning-based methods (Kushman et al., 2014; Mitra and Baral, 2016; Roy and Roth, 2018; Zhou et al., 2015; Huang et al., 2016) and semantic parsing-based methods (Shi et al., 2015; Koncel-Kedziorski et al., 2015; Roy and Roth, 2015; Huang et al., 2017), these methods do not need hand-crafted features and achieve high performance on large datasets. However, they lack in capturing the specific MWPs features, which are an evidently vital component in solving MWP. More related work and feature-related information can be found in Zhang et al. (2018).

Inspired by recent work on modeling locality using multi-head attention (Li et al., 2018; Yang et al., 2018, 2019), we introduce a group attention that contains different attention mechanisms to extract various types of MWPs features. More explicitly, there are four kinds of attention mechanisms: 1) Global attention to grab global information; 2) Quantity-related attention to model the relations between the current quantity and its neighbor-words; 3) Quantity-pair attention to acquire the relations between quantities; 4) Question-related attention to capture the connections between the question and quantities. The experimental results show that the proposed model establishes the state-of-the-art performance

* corresponding author

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 6162–6167
Florence, Italy, July 28 - August 2, 2019.

on both Math23K and MAWPS datasets. In addtion, we release the source code of our model in Github[1].

2 Background: Self-Attention Network

Self-attention networks have shown impressive results in various natural language processing tasks, such as machine translation (Vaswani et al., 2017; Shaw et al., 2018) and natural language inference (Shen et al., 2018) due to their flexibility in parallel computation and power of modeling long dependencies. It can model pairwise relevance by calculating attention weights between pairs of elements of an input sequence. In Vaswani et al. (2017), they propose a self-attention computation module, known as "Scaled Dot-Product Attention"(SDPA). It is used as the basic unit of multi-head attention. This module's input contains query matrix $Q \in \mathrm{R}^{\mathrm{m}\times\mathrm{d_k}}$, key matrix $K \in \mathrm{R}^{\mathrm{m}\times\mathrm{d_k}}$ and value matrix $V \in \mathrm{R}^{\mathrm{m}\times\mathrm{d_v}}$, where m is the number of input tokens, d_k is the dimension of query or key vector, d_v is the dimension of value vector. Output can be computed by:

$$\mathrm{head} = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V, \qquad (1)$$

As Vaswani et al. (2017) found, performing attention by projecting the queries, keys, and values into subspace with different learnable projection functions instead of a single attention can enhance the capacity to capture various context information. More specifically, this attention model first transforms Q, K, and V into $\{Q_\mathrm{h}, K_\mathrm{h}, V_\mathrm{h}\}$ with weights $\{W^\mathrm{h}_Q, W^\mathrm{h}_K, W^\mathrm{h}_V\}$, and then obtains the output features $\{\mathrm{head}_1, \mathrm{head}_2, \cdots, \mathrm{head}_\mathrm{k}\}$ by SDPA, where k is the number of SDPA modules. Finally, these output features are concatenated and projected to produce the final output state $O^{'}$.

3 Approach

In this section, we introduce how the proposed framework works and the four different types of attention we designed.

3.1 Overview

We propose a sequence-to-sequence (SEQ2SEQ) model with group attention to capture different types of features in MWPs. The SEQ2SEQ model takes the text of the whole problem as the input and corresponding equation as the output. Specifically, the group attention consists of four different types of multi-head attention modules. As illustrated in Figure 1, the pre-processed input $X = \{x_1, \cdots, x_m\}$ is transformed into $H^e = \{h^e_1, \cdots, h^e_m\}$ through Bi-LSTM. We set $Q = K = V = H^e$. The output of the group attention $O^{'}$ is produced by:

$$O^{'} = \mathrm{GroupAtt}(\mathrm{Q}, \mathrm{K}, \mathrm{V}), \qquad (2)$$

Following the same paradigm in (Vaswani et al., 2017), we add a fully-connected feed forward layer to the multi-head attention mechanism layer (i.e., group attention), and each layer is followed by a residual connection and layer normalization. Consequently, the output of group attention block O is obtained.

During decoding, we employ the pipeline in (Wang et al., 2018a). The output Y is obtained through

$$y_t = \mathrm{Softmax}(\mathrm{Attention}(h^d_t, o_j)), \qquad (3)$$

where h^d_t is the hidden state at the t-th step, o_j is the j-th state vector from the output O of the group attention block.

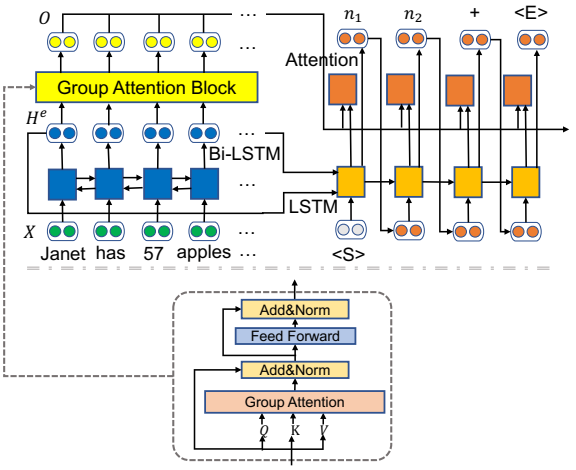

Figure 1: Framework of our approach.

3.2 Pre-Processing of MWPs

Given a MWP P and its corresponding groud-truth equation, we project words of the MWP $\{w^P_i\}^m_{i=1}$ into word embedding vectors $\{e^P_i\}^m_{i=1}$ through a word embedding matrix E, i.e., $e^P_i = Ew^P_i$. Considering the diversity of quantities in natural language, we follow the work of Wang et al. (2017) which proposed to map quantities

[1] https://github.com/lijierui/group-attention

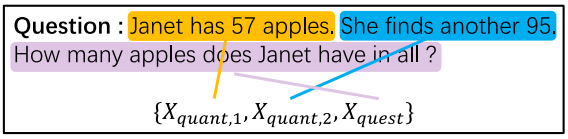

Figure 2: Example for how to separate MWPs.

into special tokens in the problem text by the following two rules: 1) All the quantities that appear in the MWP are determined if they are significant quantities that will be used in the equation using Significant Number Identify (SNI); 2) All recognized significant quantities in the MWP P are mapped to a list of mapped quantity tokens $\{n_1, ..., n_l\}$ in terms of their appearance order in the problem text, where l is the number of quantities. Through the above rules, the mapped MWP text $X = \{x_1, \cdots, x_m\}$ that will be used as the input of the SEQ2SEQ model can be acquired.

In addition, the quantity tokens in the equation are also substituted according to the corresponding mapping in problem text. For example, the mapped quantity tokens and the mapped equation of the problem in Table 1 are $\{n_1 = 4, n_2 = 52, n_3 = 7\}$ and $(n_1 + n_2) \div n_3$ respectively. To address the issue that a MWP may have more than one correct solution equations (e.g., 3×2 and 2×3 are both correct equations to solve the problem "How many apples will Tom eat after 3 days if he eats 2 apples per day?"), we normalize the equations to postfix expressions following the rules in Wang et al. (2018a), ensuring that every problem is corresponding to a unique equation. Thus, we can obtain the mapped equation E_q that will be regarded as the target sequence.

3.3 Group Attention

With the aim of implementing group attention, as illustrated in Figure 2, we separate the problem text $X = \{x_1, \cdots, x_m\}$ into quantity spans $X_{quant} = \{X_{quant,1}, \cdots, X_{quant,l}\}$ and the question span X_{quest}. The quantity span includes one or more quantity and their neighborhood words, and the question span consists of words of the question. For simplicity, the spans are separated by commas and periods, which naturally separate the sentence semantically and each span often contains one quantity, and spans with quantity (but not last) are considered as quantity spans while the last span is considered as question span since it always contains the question. By doing this, spans do not overlap with each other.

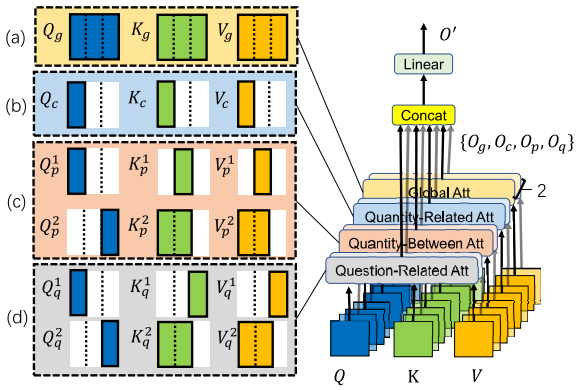

Figure 3: Group attention: (a) Global attention; (b) Quantity-related attention; (c) Quantity-pair attention; (d) Question-related attention.

As illustrated in Figure 3, following how the problem text is divided, $\{Q, K, V\}$ are masked into the input of group attention, $\{Q_g, K_g, V_g\}$, $\{Q_c, K_c, V_c\}$, $\{Q_p, K_p, V_p\}$ and $\{Q_q, K_q, V_q\}$, where g, c, p, and q are the notations of global, quantity-related, quantity-pair and question-related attention. After that, $\{O_g, O_c, O_p, O_q\}$ are computed by different groups of SDPA modules. The output of group attention O is produced by concatenating and projecting again:

$$O^{'} = \mathrm{Concat}(\mathrm{O_g}, \mathrm{O_c}, \mathrm{O_p}, \mathrm{O_q}), \qquad (4)$$

We will describe four types of group attention in detail in the following passage.

Global Attention: Document-level features play an important role in distinguishing the category of MWPs and quantities order in equations. To capture these features from a global perspective, we introduce a type of attention named as global attention, which computes the attention vector based on the whole input sequence.

For Q_g, K_g, and V_g, we set them to H^e. The output O_g can be obtained by SDPA modules belonging to global attention. For example, the word "apple" illustrated in Figure 2 will attend to the words in the whole problem text from "Janet" to "?".

Quantity-Related Attention: The words around quantity usually provide beneficial clues for MWPs solving. Hence, we introduce quantity-related attention, which focuses on the question span or quantities span where the current quantity resides.

For i-th span, its Q_c, K_c, and V_c are all derived from $X_{quant,i}$ within its own part. For example, as illustrated in Figure 2, the word "she" only attends to the words in the 2-nd quantity span "She finds another 95,".

Quantity-Pair Attention: The relationship between two quantities is of great importance in determining their associated operator. We design an attention module called quantity-pair attention, which is used to model this relationship between quantities.

The question span can be viewed as the quantity span containing an unknown quantity. Thus, the computation process consists of two parts: 1) Attention between quantities: the query Q_p is derived from $X_{quant,i}$, and corresponding K_p and V_p are stemmed from $X_{quant,j}(j \neq i)$. For example, as illustrated in Figure 2, the word "has" in the 1-st quantity span can only attend to words from the 2-nd quantity span; 2) Attention between quantities and question: the query Q_p is originated X_{quest} within the question span, and corresponding K_p and V_p are derived from X_{quant}. For example, as illustrated in Figure 2, the word "How" attends to the words in the quantity spans from "Janet" to "95,".

Question-Related Attention: The question can also derive distinguishing information such as whether the answer value is positive. Thus, we propose question-related attention, which is utilized to model the connections between question and problem description stem.

There are also two parts when modeling this type of relation: 1) Attention for quantity span: the query Q_q is derived from $X_{quant,i}$, the corresponding K_q and V_q are stemmed from X_{quest}. For example, as illustrated in Figure 2, the word "apples" in quantity span only attends to the words from the question span; 2) Attention for question span: for the query Q_q corresponding to X_{quest}, the corresponding K_q and V_q are extracted according to X_{quant}. For example, as illustrated in Figure 2, the word "does" in question span attends to the words in all the quantity spans.

4 Experiment

4.1 Experimental Setup

We evaluate the proposed model on these datasets, Math23K (Wang et al., 2017) and MAWPS (Koncel-Kedziorski et al., 2016).

Datasets: Math23K is collected from multiple online educational websites. This dataset contains 23,162 Chinese elementary school level MWPs. MAWPS is another large scale dataset which owns 2,373 arithmetic word problems after harvesting ones with a single unknown variable.

Evaluation Metrics: We use answer accuracy to evaluate our model. The accuracy calculation follows a simple formula. If a generated equation produces an answer equal to the corresponding ground truth answer, we consider it to be right.

Implementation details: For Math23K, we follow the training and test set released by (Wang et al., 2017), and we also evaluate our proposed method with 5-fold cross-validation in main results table. We adopt the pre-trained word embeddings with dimension set to 128 and use a two-layer Bi-LSTM with 256 hidden units and a group attention with four different functional 2-head attention as the encoder, and a two-layer LSTM with 512 hidden units as the decoder. Dropout probabilities for word embeddings, LSTM and group attention are all set to 0.3. The number of epochs and mini-batch size are set to 300 and 128 respectively. As to the optimizer, we use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $e = 10^{-9}$. Refer to (Vaswani et al., 2017), we use the same policy to vary the learning rate with $warmup\_steps$=2000. For MAWPS, we use 5-fold cross-validation, and the parameter setting is similar to those on Math23K.

Baselines: We compare our approach with retrieval models, deep learning based solvers. The retrieval models Jaccard and Cosine in (Robaidek et al., 2018) find the most similar math word problem in training set under a distance metric and use its equation template to compute the result. DNS (Wang et al., 2017) first applies a vanilla SEQ2SEQ model with GRU as encoder and LSTM as the decoder to solve MWPs. In (Wang et al., 2018a), the authors apply Bi-LSTM with equation normalization to reinforce the vanilla SEQ2SEQ model. T-RNN (Wang et al., 2019) launches a two-stage system named as T-RNN that first predicts a tree-structure template to be filled, and then accomplishes the template with operators predicted by the recursive neural network. In S-Aligned (Chiang and Chen, 2019), the encoder is designed to understand the semantics of problems, and the decoder focuses on deciding which symbol to generate next over semantic meanings of the generated symbols.

4.2 Main Results

	MAWPS	Math23K	Math23K*
Jaccard	45.6	-	47.2
Cosine	38.2	-	23.8
DNS	59.5	-	58.1
Bi-LSTM	69.2	66.7	-
T-RNN	66.8	66.9	-
S-Aligned	-	-	65.8
GROUP-ATT	**76.1**	**69.5**	**66.9**

Table 2: Model comparison. Notice that Math23K means the open training-test split and Math23K* means 5-fold cross-validation.

As illustrated in Table 2, we can see that retrieval approaches work poorly on both two datasets. Our method named as GROUP-ATT performs substantially better than existing deep learning based methods, increasing the accuracy from 66.9% to 69.5% on Math23K based on training-test split, from 65.8% to 66.9% on Math23K with 5-fold cross-validation and from 69.2% to 76.1% on MAWPS. In addition, DNS and T-RNN also boost the performance by integrating with retrieval methods, while (Wang et al., 2018a) improves the performance by combining different SEQ2SEQ models. However, we only focus on improving the performance of single model. It is worth noting that GROUP-ATT also achieves higher accuracy than the state-of-the-art ensemble models (Wang et al., 2019) (68.7% on Math23K based on training-test split, 67.0% on MAWPS).

	Math23K
Bi-LSTM	66.7
w/ Global Attention	68.2
w/ Quantity-Related Attention	68.2
w/ Quantity-Pair Attention	67.7
w/ Question-Related Attention	68.1

Table 3: The ablation study to quantify the role of each type of attention in group attention.

In addition, we perform an ablation study to empirically examine the ability of designed group attentions. We adopt the same parameter settings as GROUP-ATT while applying a single kind of attention with 8 heads. Table 3 shows the results of ablation study on Math23K. Although each specified attention tries to catch related information alone, it still outperforms Bi-LSTM by a margin from 1.0% to 1.5%, showing its effectiveness.

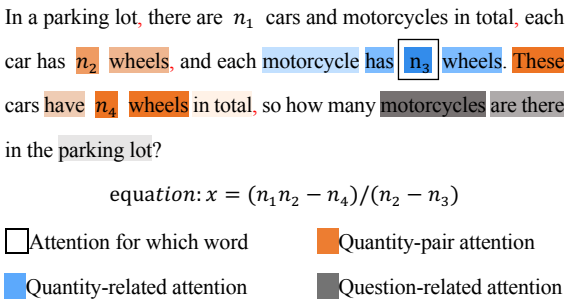

Figure 4: An example of attention visualization

4.3 Visualization Analysis of Attention

To better understand how the group attention mechanism works, we implement an attention visualization on a typical example from Math23K. As shown in Figure 4, n_3 describes how many wheels a motorcycle has. Through quantity-pair and quantity-related attention heads, n_3 pays attention to all quantities that describe the number of wheels. Question-related attention helps n_3 attend to "motorcycle" in question. In addition, surprisingly, in the quantity-pair heads, the attention of n_3 becomes more focused on the words "These", "in total" from "These vehicles have n_4 wheels in total". This indicates part-whole relation(i.e., one quantity is part of a larger quantity), mentioned in (Mitra and Baral, 2016; Roy and Roth, 2018), which is of great importance in MWPs solving. Our analysis illustrates that the hand-crafted grouping can force the model to utilize distinct information and relations conducive to solving MWPs.

5 Conclusion

In this paper, we introduce a group attention method which can reinforce the capacity of model to grab various types of MWPs specific features. We conduct experiments on two benchmarks and show significant improvements over a collection of competitive baselines, verifying the value of our model. Plus, our ablation study demonstrates the effectiveness of each group attention mechanism.

References

D. Bobrow. 1964. Natural language input for a computer problem solving system. In *Semantic information processing*, pages 146–226. MIT Press.

Ting-Rui Chiang and Yun-Nung Chen. 2019. Semantically-aligned equation generation for

solving and reasoning math word problems. In *NAACL-HLT*.

Edward A. Feigenbaum and Julian Feldman. 1963. *Computers and Thought*. McGraw-Hill, Inc., New York, NY, USA.

Danqing Huang, Jing Liu, Chin-Yew Lin, and Jian Yin. 2018a. Neural math word problem solver with reinforcement learning. In *COLING*, pages 213–223.

Danqing Huang, Shuming Shi, Chin-Yew Lin, and Jian Yin. 2017. Learning fine-grained expressions to solve math word problems. In *EMNLP*, pages 805–814.

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.

Danqing Huang, Jin-Ge Yao, Chin-Yew Lin, Qingyu Zhou, and Jian Yin. 2018b. Using intermediate representations to solve math word problems. In *ACL*, pages 419–428.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *TACL*, 3:585–597.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *NAACL*, pages 1152–1157.

Nate Kushman, Luke Zettlemoyer, Regina Barzilay, and Yoav Artzi. 2014. Learning to automatically solve algebra word problems. In *ACL*, pages 271–281.

Jian Li, Zhaopeng Tu, Baosong Yang, Michael R. Lyu, and Tong Zhang. 2018. Multi-head attention with disagreement regularization. In *EMNLP*, pages 2897–2903.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *ACL*, pages 158–167.

Arindam Mitra and Chitta Baral. 2016. Learning to use formulas to solve simple arithmetic problems. In *ACL*.

Benjamin Robaidek, Rik Koncel-Kedziorski, and Hannaneh Hajishirzi. 2018. Data-driven methods for solving algebra word problems. *CoRR*.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *EMNLP*, pages 1743–1752.

Subhro Roy and Dan Roth. 2018. Mapping to declarative knowledge for word problem solving. *TACL*, 6:159–172.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *NAACL-HLT*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI*.

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *EMNLP*, pages 1132–1142.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018a. Translating a math word problem to an expression tree. In *EMNLP*.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018b. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *AAAI*.

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 *AAAI*.

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

Baosong Yang, Jian Li, Derek F. Wong, Lidia S. Chao, Xing Wang, and Zhaopeng Tu. 2019. Context-aware self-attention networks. *CoRR*, abs/1902.05766.

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. 2018. Modeling localness for self-attention networks. In *EMNLP*, pages 4449–4458.

Dongxiang Zhang, Lei Wang, Nuo Xu, Bing Tian Dai, and Heng Tao Shen. 2018. The gap of semantic parsing: A survey on automatic math word problem solvers. *arXiv preprint arXiv:1808.07290*.

Lipu Zhou, Shuaixiang Dai, and Liwei Chen. 2015. Learn to solve algebra word problems using quadratic programming. In *EMNLP*, pages 817–822.