Translating a Math Word Problem to a Expression Tree

Lei Wang[1]*, Yan Wang[2], Deng Cai[23]*, Dongxiang Zhang[1], Xiaojiang Liu[2]

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

[3]Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong

demolei@outlook.com, {bradenwang, kieranliu}@tencent.com,
thisisjcykcd@gmail.com, zhangdo@uestc.edu.cn

Abstract

Sequence-to-sequence (SEQ2SEQ) models have been successfully applied to automatic math word problem solving. Despite its simplicity, a drawback still remains: a math word problem can be correctly solved by more than one equations. This non-deterministic transduction harms the performance of maximum likelihood estimation. In this paper, by considering the uniqueness of expression tree, we propose an equation normalization method to normalize the duplicated equations. Moreover, we analyze the performance of three popular SEQ2SEQ models on the math word problem solving. We find that each model has its own specialty in solving problems, consequently an ensemble model is then proposed to combine their advantages. Experiments on dataset Math23K show that the ensemble model with equation normalization significantly outperforms the previous state-of-the-art methods.

1 Introduction

Developing computer systems to automatically solve math word problems (MWPs) has been an interest of NLP researchers since 1963 (Feigenbaum et al., 1963; Bobrow, 1964). A typical MWP is shown in Table 1. Readers are asked to infer how many pens and pencils Jessica have in total, based on the textual problem description provided. Statistical machine learning-based methods (Kushman et al., 2014; Amnueypornsakul and Bhat, 2014; Zhou et al., 2015; Mitra and Baral, 2016; Roy and Roth, 2018) and semantic parsing-based methods (Shi et al., 2015; Koncel-Kedziorski et al., 2015; Roy and Roth, 2015; Huang et al., 2017) are proposed to tackle this problem, yet they still require considerable manual efforts on feature or template designing. For more literatures about solving math word problems automatically, refer to a recent survey paper Zhang et al. (2018).

Problem: Dan has 5 pens and 3 pencils, Jessica has 4 more pens and 2 less pencils than him. How many pens and pencils does Jessica have in total?
Equation: $x = 5+4+3-2$; **Solution:** 10

Table 1: A math word problem

Recently, the Deep Neural Networks (DNNs) have opened a new direction towards automatic MWP solving. Ling et al. (2017) take multiple-choice problems as input and automatically generate rationale text and the final choice. Wang et al. (2018) then make the first attempt of applying deep reinforcement learning to arithmetic word problem solving. Wang et al. (2017) train a deep neural solver (DNS) that needs no hand-crafted features, using the SEQ2SEQ model to automatically learn the problem-to-equation mapping.

Although promising results have been reported, the model in (Wang et al., 2017) still suffers from an equation duplication problem: a MWP can be solved by multiple equations. Taking the problem in Table 1 as an example, it can be solved by various equations such as $x = 5 + 4 + 3 - 2$, $x = 4 + (5 - 2) + 3$ and $x = 5 - 2 + 3 + 4$. This duplication problem results in a non-deterministic output space, which has a negative impact on the performance of most data-driven methods. In this paper, by considering the uniqueness of expression tree, we propose an equation normalization method to solve this problem.

Given the success of different SEQ2SEQ models on machine translation (such as recurrent encoder-decoder (Wu et al., 2016), Convolutional SEQ2SEQ model (Gehring et al., 2017) and Trans-

*The work was done when Lei Wang and Deng Cai were interns at Tencent AI Lab.

former (Vaswani et al., 2017)), it is promising to adapt them to MWP solving. In this paper, we compare the performance of three state-of-the-art SEQ2SEQ models on MWP solving. We observe that different models are able to correctly solve different MWPs, therefore, as a matter of course, an ensemble model is proposed to achieve higher performance. Experiments on dataset Math23K show that by adopting the equation normalization and model ensemble techniques, the accuracy boosts from 60.7% to 68.4%.

The remaining part of this paper is organized as follows: we first introduce the SEQ2SEQ Framework in Section 2. Then the equation normalization process is presented in Section 3, following which three SEQ2SEQ models and an ensemble model are applied to MWP solving in Section 4. The experimental results are presented in Section 5. Finally we conclude this paper in Section 6.

2 SEQ2SEQ Framework

The process of using SEQ2SEQ model to solve MWPs can be divided into two stages (Wang et al., 2017). In the first stage (number mapping stage), significant numbers (numbers that will be used in real calculation) in problem P are mapped to a list of number tokens $\{n_1, \dots, n_m\}$ by their natural order in the problem text. Throughout this paper, we use the significant number identification (SNI) module proposed in (Wang et al., 2017) to identify whether a number is significant. In the second stage, SEQ2SEQ models can be trained by taking the problem text as the source sequence and equation templates (equations after number mapping) as the target sequence.

Taking the problem P in Table 1 as an example, first we can obtain a number mapping $M : \{n_1 = 5; \quad n_2 = 3; \quad n_3 = 4; \quad n_4 = 2;\}$, and transform the given equation $E_P : x = 5+4+3-2$ to an equation template $T_P : x = n_1+n_3+n_2-n_4$. During training, the objective of our SEQ2SEQ model is to maximize the conditional probability $P(T_p|P)$, which will be decomposed to token-wise probabilities. During decoding, we use beam search to approximate the most likely equation template. After that, we replace the number tokens with actual numbers and calculate the solution S with a math solver.

3 Equation Normalization

In the number mapping stage, the equations E_P have been successfully transformed to equation templates T_P. However, due to the equation duplication problem introduced in Section 1, this problem-equation templates formalization is a non-deterministic transduction that will have adverse effects on the performance of maximum likelihood estimation. There are two types of equation duplication: 1) order duplication such as "$n1 + n3 + n2$" and "$n1 + n2 + n3$", 2) bracket duplication such as "$n_1 + n_3 - n_2$" and "$n1 + (n_3 - n_2)$".

To normalize the order-duplicated templates, we define two normalization rules:

- Rule 1: Two duplicated equation templates with unequal length should be normalized to the shorter one. For example, two equation templates "$n_1 + n_2 + n_3 + n_3 - n_3$", "$n_1 + n_2 + n_3$" should be normalized to the latter one.

- Rule 2: The number tokens in equation templates should be ordered as close as possible to their order in number mapping. For example, three equation templates "$n_1 + n_3 + n_2$", "$n_1 + n_2 + n_3$" and "$n_3 + n_1 + n_2$" should be normalized to "$n_1 + n_2 + n_3$".

To solve the bracket duplication problem, we further normalize the equation templates to an expression tree. Every inner node in the expression tree is an operator with two children, while each leaf node is expected to be a number token. An example of expressing equation template $n_1 + n_2 + n_3 - n_4$ as the unique expression tree is shown in Figure 1.

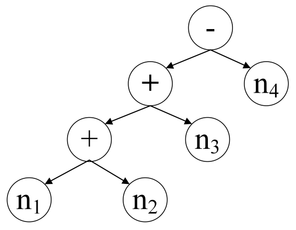

Figure 1: A Unique Expression Tree

After equation normalization, the SEQ2SEQ models can solve MWPs by taking problem text as source sequence and the postorder traversal of an unique expression tree as target sequence, as shown in Figure 2.

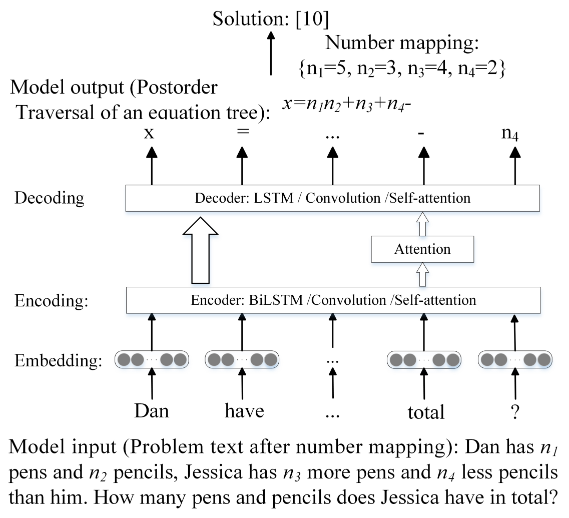
Solution: [10]
Number mapping: {n1=5, n2=3, n3=4, n4=2}
Model output (Postorder Traversal of an equation tree):
x=n1n2+n3+n4-
x = ... - n4
Decoding
Decoder: LSTM / Convolution /Self-attention
Attention
Encoding:
Encoder: BiLSTM /Convolution /Self-attention
Embedding:
Dan have ... total ?
Model input (Problem text after number mapping): Dan has n1 pens and n2 pencils, Jessica has n3 more pens and n4 less pencils than him. How many pens and pencils does Jessica have in total?

Figure 2: Framework of SEQ2SEQ models

4 Models

In this section, we present three types of SEQ2SEQ models to solve MWPs: bidirectional Long Short Term Memory network (BiLSTM) (Wu et al., 2016), Convolutional SEQ2SEQ model (Gehring et al., 2017), and Transformer (Vaswani et al., 2017). To benefit the output accuracy with all three architectures, we propose to use a simple ensemble method.

4.1 BiLSTM

The BiLSTM model uses two LSTMs (forward and backward) to learn the representation of each token in the sequence based on both the past and the future context of the token. At each time step of decoding, the deocder uses a global attention mechanism to read those representations.

In more detail, we use two-layer Bi-LSTM cells with 256 hidden units as encoder, and two layers LSTM cells with 512 hidden units as decoder. In addition, we use Adam optimizer with learning rate $1e^{-3}$, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. The epochs, minibatch size, and dropout rate are set to 100, 64, and 0.5, respectively.

4.2 ConvS2S

ConvS2S (Gehring et al., 2017) uses a convolutional architecture instead of RNNs. Both encoder and decoder share the same convolutional structure that uses n kernels striding from one side to the other, and uses gate linear units as non-linearity activations over the output of convolution.

Our ConvS2S model adopts a four layers encoder and a three layers decoder, both using kernels of width 3 and hidden size 256. We adopt early stopping and learning rate annealing and set max-epochs equals to 100.

4.3 Transformer

Vaswani et al. (2017) proposed the Transformer based on an attention mechanism without relying on any convolutional or recurrent architecture. Both encoder and decoder are composed of a stack of identical layers. Each layer contains two parts: a multi-head self-attention module and a position-wise fully-connected feed-forward network.

Our transformer is four layers deep, with $n_{head} = 16$, $d_k = 12$, $d_v = 32$, and $d_{model} = 512$, where n_{head} is the number of heads of its self-attention, d_k is the dimension of keys, d_v is the dimension of values, and d_{model} is the output dimension of each sub-layer. In addition, we use Adam optimizer with learning rate $1e^{-9}$, $\beta_1 = 0.9$, $\beta_2 = 0.99$, and dropout rate of 0.3.

4.4 Ensemble Model

Through careful observation (detailed in Section 5.2), we find that each model has a speciality in solving problems. Therefore, we propose an ensemble model which selects the result according to models' generation probability:

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t | y_{<t}, \mathbf{x})$$

where $\mathbf{y} = \{y_1, ..., y_T\}$ is the target sequence, and $\mathbf{x} = \{x_1, ..., x_S\}$ is the source sequence. Finally, the output of the model with the highest generation probability is selected as the final output.

5 Experiment

In this section, we conduct experiments on dataset Math23K to examine the performance of different SEQ2SEQ models. Our main experimental result is to show a significant improvement over the baseline methods. We further conduct a case study to analyze why different SEQ2SEQ models can solve different kinds of MWPs.

Dataset: Math23K[1] collected by Wang et al. (2017) contains 23,162 labeled MWPs. All these

[1]https://ai.tencent.com/ailab/Deep_Neural_Solver_for_Math_Word_Problems.html

	Acc w/o EN (%)	Acc w/ EN (%)
DNS	58.1	60.7
Bi-LSTM	59.6	66.7
ConvS2S	61.5	64.2
Transformer	59.0	62.3
Ensemble	66.4	**68.4**

Table 2: Model comparison. EN is short for equation normalization

	Bi-LSTM	Transformer	ConvS2S
w/o EN	59.6	59.0	61.5
+ SE	63.1	59.9	62.2
+ OE	63.7	60.7	62.9
+ EB	65.3	61.2	62.9

Table 3: The ablation study of three equation normalization methods. SE is the first Rule mentioned in Section 3. OE is the second rule mentioned in Section 3. EB means eliminating the brackets.

problems are linear algebra questions with only one unknown variable.

Baselines: We compare our methods with two baselines: DNS and DNS-Hybrid. Both of them are proposed in (Wang et al., 2017), with state-of-the-art performance on dataset Math23K. The DNS is a vanilla SEQ2SEQ model that adopts GRU (Chung et al., 2014) as encoder and LSTM as decoder. The DNS-Hybrid is a hybrid model that combines DNS and a retrieval-based solver to achieve better performance.

5.1 Results

In experiments, we use the testing set in Math23K as the test set, and randomly split 1, 000 problems from the training set as validation set. Evaluation results are summarized in Table 2. First, to examine the effectiveness of equation normalization, model performance with and without equation normalization are compared. Then the performance of DNS, DNS-Hybrid, Bi-LSTM, ConvS2S, Transformer, and Ensemble model are examined on the dataset.

Several observations can be made from the results. First, the equation normalization process significantly improves the performance of each model. The accuracy of different models gain increases from 2.7% to 7.1% after equation normalization. Second, Bi-LSTM, ConvS2S, Transformer can achieve much higher performance than DNS, which means that popular machine translation models are also efficient in automatic MWP solving. Third, by combining the SEQ2SEQ models, our ensemble model gains additional 1.7% increase on accuracy.

In addition, we have further conducted three extra experiments to disentangle the benefits of three different EN techniques. Table 3 gives the details of the ablation study of the three SEQ2SEQ models. Taking Bi-LSTM as an example, accuracies of rule 1 (SE), rule 2 (OE) and eliminating brackets (EB) are 63.1%, 63.7% and 65.3%, respectively. Obviously, the performance of SEQ2ESQ models benefits from the equation normalization technologies.

5.2 Case Study

Further, we conduct a case analysis on the capability of different SEQ2SEQ models and provide three examples in Table 4. Our analysis is summarized as follows: 1) Transformer occasionally generates mathematically incorrect templates, while Bi-LSTM and ConvS2S almost do not, as shown in Example 1. This is probably because the size of training data is still not enough to train the multi-head self-attention structures; 2) In Example 2, the Transformer is adapted to solve problems that require complex inference. It is mainly because different heads in a self-attention structure can model various types of relationships between number tokens; 3) The multi-layer convolutional block structure in ConvS2S can properly process the context information of number tokens. In Example 3, it is the only one that captures the relationship between stamp A and stamp B.

6 Conclusion

In this paper, we first propose an equation normalization method that normalizes duplicated equation templates to an expression tree. We test different SEQ2SEQ models on MWP solving and propose an ensemble model to achieve higher performance. Experimental results demonstrate that the proposed equation normalization method and the ensemble model can significantly improve the state-of-the-art methods.

Acknowledgement

This work is supported in part by the National Nature Science Foundation of China under grants No. 61602087, and the Fundamental Research Funds for the Central Universities under grants No. ZYGX2016J080.

Example 1: Two biological groups have produced 690 (n_1) butterfly specimens in 15 (n_2) days. The first group produced 20 (n_3) each day. How many did the second group produced each day? **Bi-LSTM:** $n_1 n_2 / n_3 -$; (correct) **ConvS2S:** $n_2 n_3 + n_1 *$; (error) **Transformer:** $n_2 n_1 n_3 n_3 +$; (error)
Example 2: A plane, in a speed of 500 (n_1) km/h, costs 3 (n_2) hours traveling from city A to city B. It only costs 2 (n_3) hours for return. How much is the average speed of the plane during this round-trip? **Bi-LSTM:** $n_1 n_2 * n_3 *$; (error) **ConvS2S:** $11 + 1 n_1 / 1 n_2 / + /$; (error) **Transformer:** $n_1 n_2 * n_3 * n_2 n_3 + /$; (correct)
Example 3: Stamp A is 2 (n_1) paise denomination, and stamp B is 7 (n_2) paise denomination. If we are asked to buy 10 (n_3) of each, how much more does it cost to buy stamps A than to buy stamps B. **Bi-LSTM:** $n_1 n_2 n_3 * -$; (error) **ConvS2S:** $n_1 n_3 * n_2 n_3 * -$; (correct) **Transformer:** $n_2 n_2 * n_2 n_3 * -$; (error)

Table 4: Three examples of solving MWP with SEQ2SEQ model. Note that the results are postorder traversal of expression trees, and the problems are translated to English for brevity.

References

Bussaba Amnueypornsakul and Suma Bhat. 2014. Machine-guided solution to mathematical word problems. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation, PACLIC 28, Cape Panwa Hotel, Phuket, Thailand, December 12-14, 2014*, pages 111–119.

Daniel G Bobrow. 1964. Natural language input for a computer problem solving system.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Edward A Feigenbaum, Julian Feldman, et al. 1963. *Computers and thought*. New York.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Danqing Huang, Shuming Shi, Chin-Yew Lin, and Jian Yin. 2017. Learning fine-grained expressions to solve math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 805–814. Association for Computational Linguistics.

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

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. Association for Computational Linguistics.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167. Association for Computational Linguistics.

Arindam Mitra and Chitta Baral. 2016. Learning to use formulas to solve simple arithmetic problems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1743–1752.

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

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1132–1142.

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 *Advances in Neural Information Processing Systems*, pages 6000–6010.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018. Mathdqn: Solving arithmeticword problems via deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

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 *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 817–822.