# GTT-Bert: Pre-training of Graph-To-Tree Model for Math Word Problems

Ruolin Dou(✉), Dong Liang(✉), Nan Wang(✉), and Junxuan Wang(✉)

Beijing University of Posts and Telecommunications, Beijing, China
{idouruolin,liangdong,wn98,Wjunxuan}@bupt.edu.cn

**Abstract.** Math word problem (MWP) is an important problem in the field of intelligent education and natural language processing. The existing models for solving MWP problems mainly include sequence to sequence (Seq2Seq), sequence to tree (Seq2Tree), graph to tree (Graph2Tree) and other methods. Graph2Tree model can well capture the relationship and order representation between quantities. However, the existing Graph2Tree model usually uses the embedded layer to represent the input text sequence as a word vector, which does not obtain the representation without paying attention to numerical attributes and context representation interpretation information. We propose a pre-train- model based on Graph2Tree structure. The experimental results show that the performance of Graph2Tree model with our pre-training model is significantly better than the existing Graph2Tree model.

**Keywords:** education · math word problem · natural language processing · pre-training model · representation learning

## 1 Introduction

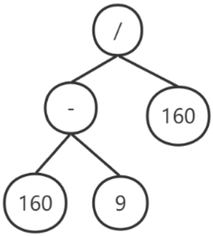
The system for automatically solving MWP problems was first proposed by Bobrow (1964). Table 1 shows an example of MWP problems. The input of the system is a text description of a mathematical problem with a real-world background, and the output is the equation expression and value output of the problem-solving variables [3]. Recent years, DNN has significantly improved the accuracy of MWP tasks. These methods can be roughly divided into the following three categories: sequence to sequence structure (Seq2Seq) based model, sequence to tree structure (Seq2Tree) based model, and graph to tree structure (Graph2Tree) based model [2].

Wang et.al. [3] first proposed to use the deep learning method to solve MWP problem, and proposed a large-scale MWP dataset. They used the Seq2Seq structure to directly complete the mapping from “problem text” to “equation”. Liang et.al [4] (2022) found that BERTGen [4] and RoBERTGen perform well on Math23K dataset (76.9%).

The Seq2tree [6, 7] based model is actually a variation of the seq2seq based model. Seq2tree model convert the expression after number mapping into a tree structure as the output of model training, Since the mathematical symbols and connection methods at the parent and child nodes are fixed, this method can effectively limit the diversity of expressions.

X. Yuan et al. (Eds.): ICEKIM 2023, AHCS 13, pp. 1151–1158, 2023.
https://doi.org/10.2991/978-94-6463-172-2_121

**Table 1.** A Math Word Problem

| | |
|---|---|
| **Problem** | In a group of 160 people , 90 have an age of more 30 years , and the others have an age of less than 20 years . if a person is selected at random from this group , what is the probability the person ' s age is less than 20 ? |
| **Equation** | x=((160-90)/160) |
| **Ans** | 0.4375 |
| **Reasoning Logic** |  |

Most models based on Seq2Seq or Seq2Tree structure are not able to capture the relationship between order information and quantities sufficiently, which will cause incorrect quantity representation and reconciliation expression. In order to solve the above problems. Zhang et.al [8] proposed the Graph2Tree model by constructing Quantity Cell Graph and Quantity Comparison Graph. This model can effectively express the quantity and position order relationship in the problem text, and the accuracy on Math23K dataset is improved to 75%.

However, the Graph2Tree model only uses the form of numerical placeholders to replace real numbers, focusing on logical reasoning, and paying less attention to reusable numerical attributes and numerical context representation information. To solve this problem, we propose a Bert coding pre-training model based on Graph2Tree structure. Our contribution is to adjust the pre-training model by introducing target tasks. The target tasks adopted are as follows:

1. The first group of pre-training models is for position information: we add the quantity and position order relationship in the text to the Bert pre-training model to encourage contextual representation to capture the number position relationship.
2. The second group of pre-training objectives aims at the regression task in the text features: we use the number of subtrees and the type of keyword operators in the question solution to enhance the local information expression coding of the text.
3. The third group of pre-training objectives is aimed at the classification task in the text features: we enhance the global text information coding by predicting the problem type.

The experimental results show that our model is superior to the existing Graph2Tree structure model.

## 2 Problem Statement

We express a math word problem as $(P, E, R)$, where $P$ represents the problem text sequence, and P is represented as the token sequence $\{p_1, p_2, ..., p_n\}$ by word segmentation. The equation expression can be reconstructed by Reverse Polish notation rules, so E can be expressed as a sequence $E = \{e_1, e_2, ..., e_m\}$ without parentheses. By element type division, the elements in $E$ can be divided into operator set $O$, and the number set that exists in the answer but does not exist in the text of the question, that is, constant set $N$. The set of numbers that exist in the answer and in the text of the question is the set of known conditions $K$. $R$ represents the operation result.

## 3 Materials and Methods

The goal of the pre-training model based on Graph2Tree structure is to use number features and topic category features from problem text as representation learning constraints. As shown in Fig. 1, in the coding process, the pre-training model adjusts itself by using the position information of the known quantity of related words, the quantity information in the text features of the topic, and the classification information of the topic category.

### 3.1 Graphs

We construct quantity cells set for each problem, $Q = \{q_1, q_2, ..., q_m\} \in P$. For each element in $Q$, we use the dependency parser to construct the corresponding related nouns set $S_{Ni} \in P$, related verbs set $S_{Vi} \in P$, related adjective set $S_{Ai} \in P$. Combine the corresponding sets of each quantity cells to get related nouns set $S_N = \{S_{N1}, S_{N2}, ..., S_{Nm}\}$, related verbs set $S_V = \{S_{V1}, S_{V2}, ..., S_{Vm}\}$ and related adjective set $S_A = \{S_{A1}, S_{A2}, ..., S_{Am}\}$ of a problem.

The following 3 subgraphs will input into 3 GCN, and the output of each GCN is connected to form the output of graph $G$.

**1) Associated Nouns Graph**
The adjacency graph formed by quantity cells set $Q$ and its corresponding set of nouns $S_N$.

**2) Associated Verbs Graph**
The adjacency graph formed by quantity cells set $Q$ and its corresponding set of verbs $S_{\mathrm{V}}$.

**3) Associated Adjectives Graph**
The adjacency graph formed by quantity cells set $Q$ and its corresponding set of verbs $S_A$.

The graph-based topic number representation and location relationship are preliminarily constructed through the above graphs. Since the graph convolution network uses Laplace matrix to extract the spatial features of the topology graph, the K-head graph is

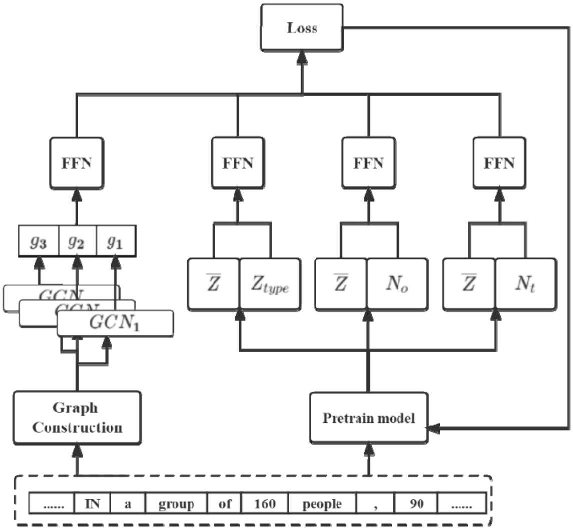


**Fig. 1.** Pre-training model

used to learn the node features of each graph. As shown in Formula 1, for the parameter matrix $W$ of a graph, the output of GCN is as shown in Formula 1:

$$GCN_k(X, A_k) = \mathrm{softmax}\left(A_\mathrm{k}\mathrm{ReLU}\left(A_k X W^{(0)}\right) W^{(1)}\right) \quad (1)$$

In Formula 1, k $\in (1, K)$ and $X$ represents the feature matrix of the input features and the deep hidden state.

Connect the output of $K$ graphs to get the output value

$$Z_g = con\mathrm{cat}(GCN_1(A_1, H), GCN_2(A_2, H), ..., GCN_k(A_k, H)) \quad (2)$$

A two-layer feed-forward neural network, a layer-norm layer, and a residual connection are used to enhance the K-header convolution network, resulting in $\mathrm{Z}_1$. As shown in Formula 3, node characteristics are aggregated into a graph using a pooling operation as output.

$$z_g = FC(MinPool(Z_1)) \quad (3)$$

Add the loss function shown in Formula 4 to the pre-training model:

$$L_{graphpre} = MSE\left(FFN(X), z_g\right) \quad (4)$$

MSE represents mean square error and $FFN$ represents a feed forward neural network consisting of two fully connected layers and a ReLU activation function.

## 3.2 Quantity Information Related to the Problem

### 1) Tree Counting

In a math word problem, each subtree represents a step in the solution. We use the number of subtrees as the fine-tuning index of the pre-training model, the loss function shown in Formula 5 is added to the pre-training model:

$$L_{TreeCount} = MSE\left(FFN\left(\overline{Z}\right), N_t\right) \tag{5}$$

where Z represents the mean vector of the output Z of the text encoder:

$$\mathrm{Z} = \mathrm{encoder(P)} \tag{6}$$

**2) Number of Operator Types**

The number of operation pairs with different operation relationships in a single problem reflects the logical complexity of the problem to a certain extent. Add loss function shown in Formula 7 to the pre-training model:

$$L_{opCount} = MSE\left(FFN\left(\overline{Z}\right), N_o\right) \tag{7}$$

### 3.3 Problem Type

Problems of the same type have similar text characteristics and problem-solving templates. Based on predefined attribute grammar, Hong et.al [9] (2021) classify the existing problem text into four problem types: Task, Motion, Relation and Price. Add problem type prediction tasks to the pre-training model to get the global characteristics of the problem text:

$$L_{TypePred} = CE\left(FFN\left(\overline{Z}\right), y_s\right) \tag{8}$$

where CE represents the cross-entropy loss function for classification tasks.

Therefore, the overall loss function of the pre-training model is

$$L = L_{model} + L_{graphpre} + L_{TreeCount} + L_{opCount} + L_{TypePred} \tag{9}$$

## 4 MWP Solving

The structure of the MWP-solving model is shown in the Fig. 2. The MWP-solving model based on graph structure first uses the pre-training model to semantically characterize MWP text, and uses GRU to extract text features. The output is represented as a node. Simultaneously, we construct Associated Nouns Graph, Associated Verbs Graph and Associated Adjectives Graph. These figures were inputted into GCNs to learn characteristics of icon recognition. The pooling layer aggregates all nodes into a graph embedding vector as the output of the graph converter. The final output graph represents the updated node.

### 4.1 Dataset

We used Math23K (Wang et al., 2017) dataset, which contains 23162 questions in total. The dataset contains the question text, the answer equation, and the answer value. All problems are linear algebraic problems with only one unknown quantity.

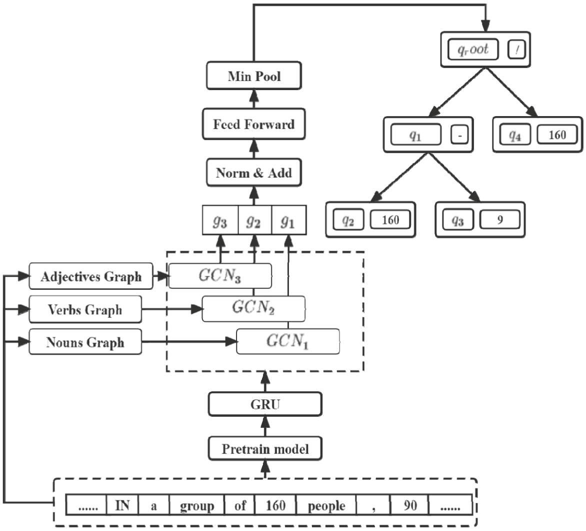


**Fig. 2.** Overview of the proposed model

### 4.2 Baseline

First, we compare our method with the classical solution DNS model of MWP problem, which adopts the Seq2Seq structure.

We also compare without using the pre-training model, Graph2Tree structure model (Zhang et al., 2020) is used as a comparison of the effectiveness of the pre-training model.

### 4.3 Model Comparison

As shown in Fig. 3, our pre-training model preforms a smaller initial loss value and a faster loss reduction rate during the first 10 epochs learning.

After training 10 epochs, the loss of our model is reduced to 0.2, and the loss of the automatic MWP solving model based on Graph2Tree structure without using the pre-training model is 0.3. DNS model loss is 0.5. Our model has faster learning speed.

As shown in Table 2, compared with the DNS model, the accuracy rate of our model equations and the accuracy rate of answers increased by 13% and 15% respectively. Compared with the Graph2Tree model that does not use our pre-training model, our model equation accuracy and answer accuracy increased by 7% and 8% respectively.

As shown in Table 3, we use three models to solve a problem in Math23K dataset. The DNS model is correct when solving problems, so the known condition “4 wheels” is not fully used. The Graph2Tree model that does not use our pre-training model ignores this condition. Our model correctly solves this problem. This example shows that our pre-training model is helpful to capture subtle changes in problem description and has a stronger ability to understand problems.

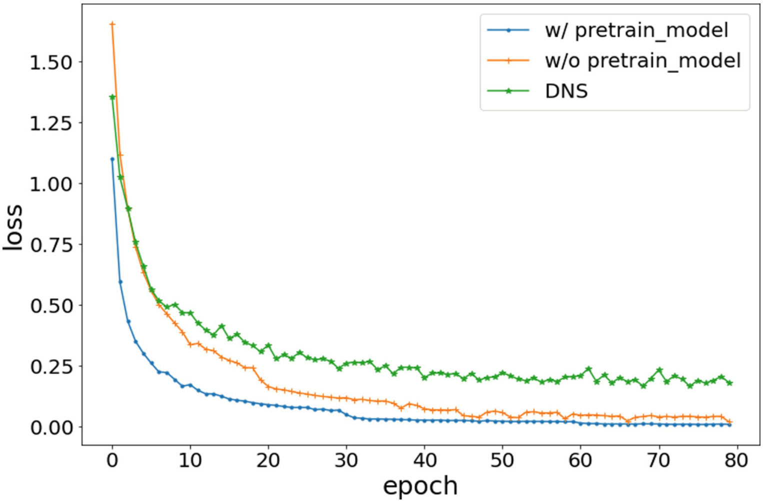


**Fig. 3.** Loss of different models

**Table 2.** Accuracy of different models.

| | **Equation accuracy** | **Answer accuracy** |
|---|---|---|
| **DNS** | 58% | 68% |
| G2T **w/o pretrained model** | 64% | 76% |
| G2T **w/pretrained model** | 71% | 84% |

**Table 3.** Comparison of results

| **Problem Text** | There are totally 48 cars and motorcycles in a parking lot. Each car has 4 wheels and each motorcycle has 3 wheels. If they have 172 wheels in total. How many motorcycles are there in the parking lot? |
|---|---|
| **DNS** | x = 48*4–172/3(✗) |
| **G2T w/o pretrained model** | x = 48−(48−172)/3(✗) |
| **G2T w/pretrained model** | x = (48∗4−172)/(4−3) |

## 5 Conclusions

We propose a pre-training model based on Graph2Tree structure, and fine tune Robert coding model by constructing three sets of loss functions for location information, regression tasks in text features, and classification tasks in text features. The experiment shows that the pre-training model proposed by us has a stronger ability to understand complex problems in Graph2Tree problem solving model.

## References

1. Sundaram S S, Gurajada S, Fisichella M, et al. Why are NLP Models Fumbling at Elementary Math? A Survey of Deep Learning based Word Problem Solvers[J]. arXiv preprint arXiv:2205.15683, 2022.
2. S. Ughade and S. Kumbhar, "Survey on mathematical word problem 1018 solving using natural language processing," in Proc. 1st Int. Conf. Innov. 1019 Inf. Commun. Technol. (ICIICT), Apr. 2019, pp. 1–5.
3. Y. Wang, X. Liu, and S. Shi, "Deep neural solver for math word problems," Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 845–854, 2017.
4. Z. Liang, J. Zhang, L. Wang, W. Qin, Y. Lan, J. Shao, X. Zhang, "MWP-BERT: Numeracy-augmented pre-training for math word problem solving," Proceedings of the 2022 Conference on Association for Computational Linguistics (NAACL 2022), pp. 997–1009, 2022.
5. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training 1072 of deep bidirectional transformers for language understanding," 2018, 1073. arXiv:1810.04805
6. L. Wang, Y. Wang, D. Cai, D. Zhang, and X. Liu, "Translating a math word 1061 problem to a expression tree," in Proc. Conf. Empirical Methods Natural 1062 Lang. Process., 2018, pp. 1064–1069.
7. Q. Liu, W Guau, S. Li, aud D. Kawahara, "Treestructured decoding for solving math word problems," Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2370–2379, 2019.
8. J. Zhang, L. Wang, R. K.-W Lee, Y. Bin, Y. Waug, J. Shao, and E.-P. Lim, "Graph-to-tree learning for solving math word problems," Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 3928–3937, 2020.
9. Y. Hong, Q. Li, R. Gong, D. Ciao, S. Huang, and S-C Zhu, "Smart: A situation model for algebra story problems via attributed grammar," The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI-21, 2021.