RESEARCH ARTICLE OPEN ACCESS

A Survey on Machine Learning Methods for Solving Elementary Math Word Problems

Xinxin Li [1], Jiawen Wang [2]
[1] School of Computer Science and Technology, Shandong University of Technology - China
[2] Cangzhou Technical College - China

ABSTRACT
Solving math word problems refers to mapping math word problems into logical forms that can be understood by machines and then obtaining the answer by reasoning or calculation. This task is a difficult issue in many areas, such as artificial intelligence, cognitive psychology, and mathematics education. In this paper, with the development of machine learning methods, we divide math word problem solving methods into rule-based, statistical machine learning, and deep learning methods. We also discuss different representation methods for the problem, including transition-based, template-based, tree-based, and semantic representation methods, and perform a comparative analysis. Finally, this paper describes current challenges and discusses future direction.

Keywords: - Math word problems, Machine learning, Deep learning

I. INTRODUCTION

With the emergence of artificial intelligence in the 1960s, research on math word problem solving began [1]. Solving math word problems (MWP) refers to mapping human-readable sentences into machine-understandable logical forms and then performing inferences to obtain answers. Unlike machine reading comprehension that selects the correct results from the given options, math word problem solving requires understanding the semantic information in the problem and obtaining the answer through reasoning. Elementary math word problem is one type of MWP that has one answer to be solved.

Along with the development of artificial intelligence and machine learning methods, the methods for solving math word problems are usually divided into three categories: rule-based methods, statistical machine learning methods, and deep learning methods. Rule-based method uses pre-defined strategies to map the problem to propositional logic, and then uses logical reasoning or arithmetic operations to obtain the results [2]-[4]. Statistical machine learning methods use support vector machines and other methods to predict the equations or mathematical operators of the problems [5]–[7]. Deep learning methods have been successful in the fields of computer vision, speech recognition, and natural language processing [8], [9]. For example, tasks such as machine reading comprehension and speech recognition have surpassed human levels, and have made progress on math word problems [10], [11]. However, elementary math word problem solvers are still facing large challenges.

Mukherjee and Garain presented a review of methods before 2009, in which most methods are rule-based methods and no recent methods are discussed [4]. Mandal and Naskar provided a detailed description of statistical machine learning methods, but missed deep learning methods [12]. Zhang et al. divided math word problems into arithmetic problems, equation problems, and geometric problems, and described the methods for each type of problems separately [2]. Unlike Zhang's work, this paper focuses on elementary math word problems and adopts a more detailed description of the representation of the problems. Recent methods are analyzed, especially pre-training models and graph neural network models.

II. PROBLEM DESCRIPTION

Solving math word problems refer to obtain the unknown quantities from give problem text. It can be described as follows: given a problem text containing k words $\mathbf{x} = w_1, \dots, w_k$, which contains m known quantities $v_1, v_2, \dots, v_m$, there are n unknown quantities $u_1, u_2, \dots, u_n$ that need to be solved. For elementary math word problems, there is only one unknown quantity. The elementary math word problem discussed in this paper refers to the problems that can be solved by arithmetic operations. They don't contain advanced operations such as square root, exponentiation, and logarithm. Only addition, subtraction, multiplication and division operations are considered.

Table 1 shows an example of an elementary math word problem. The problem contains two known quantities 3 and 5, and there is only one unknown quantity to be solved. The arithmetic method and equation method are usually used. The arithmetic method means that only known quantities and appropriate operators are in the mathematical expression, and the unknown quantity is the result of the expression. In this example, we need to find the arithmetic expression 3+5 for the problem. The equation method refers to finding the mathematical equation including known and unknown quantities. For elementary math word problem, the equation generally means linear equation with one unknown, such as the equation of this problem is x=3+5.

TABLE I
AN ELEMENTARY MATH WORD PROBLEM

Problem	Dan has 3 pens, Jessica has 5 more pens that Dan. How many pens does Jessica have?
Arithmetic expression	3+5
Equation	x=3+5
Answer	8

III. RULE-BASED METHODS

The early elementary math word problems adopted rule-based methods. The idea is to determine whether the representation of the problem (sentence sequence, syntactic structure, or semantic structure) matches the given rules. If the rules match, the problem is converted into a equation or actions. Rule-based systems include STUDENT, DEDUCOM, CARPS, and HAPPINESS, etc [13]–[16].

The earliest solver was STUDENT system in 1964, which converted the problems into logical expressions by a set of rules [13]. An example is given in table 2. The system first converted specific words in the sentence and annotated arithmetic operators, verbs, prepositions, and question marks. For example, sentence 1 in table 2 is annotated as sentence 2. Secondly, a long sentence is split into multiple independent simple sentences according to the rules. For example, sentence 3 that satisfies rule 4 is split into two sentences, where "If" is followed by sentence 5 and a comma is followed by sentence 1. Then, the system converts the simple sentences into an equation. For example, sentence 6 is converted to equation 7, and the clause without operators is converted to the unknown quantity by removing punctuation and crown, e.g. NUMBER in sentence 7. Finally, the converted equation can be solved by logical deduction. We can get the answer 49 from equation 6.

TABLE 2
STUDENT EXAMPLE

No	**Sentence/Rule**
1	What is the number of customers Tom gets ?
2	(What/QWORD) is the number (of/OP) customers Tom (gets/VERB) (QMARK/DLM)
3	If the number of customers Tom gets is 2 times the square 20 percent of the number of advertisements he runs, what is the number of customrs Tom gets ?
4	(IF \$, (\$1/ QWORD(\$))
5	The number of customers Tom gets is 2 times the square 20 percent of the number of advertisements he runs.
6	A number is added to 18. This sum is 67.
7	(EQUAL (PLUS (NUMBER) 18) 67)

DEDUCOM system made further research on logical deduction and introduced depth-first search strategy and recursive algorithm to find the answer of logical expression [14], [17]. The system divided all problems into three types: conventional calculation, simple search, and their combination. For a complex problem $q = g(q_1, q_2, \cdots, q_n)$, the solution was transformed into a smaller problem set $q_1, q_2, \cdots, q_n$. The system correctly answered 10 questions under the given 68 facts, and improved its deductive ability by adding more facts. The DEDUCOM system can't process the original problem text but only logical expressions.

Based on the STUDENT system, CARPS system introduced structured information into the problem representation, which was mainly used to solve the calculus rate problem [15], [18]. Unlike the STUDENT system, which used simple text matching, the CARPS system employed structured information. Through grammatical analysis of the question text, the information of each object was stored in a tree structure, which was more robust. Assuming that the phrases "CONICAL PILE" and "PILE OF SAND" appear in the same input question, the STUDENT system cannot know that they represent the same object, and the CARPS system can use a tree structure to store these related items.

With the development of cognitive psychology, researchers have realized that children's ability to solve math word problems is related to the conceptual knowledge that they understand. Schema is a structure that represents general concepts stored in memory that describes conceptual knowledge in math word problems. The content of schema representation includes objects, scenes, events, actions, and sets. The concept of schema theory was first introduced into psychology by Frederic Bartlett, and expanded to education field by Richard C. Anderson [19]. In artificial intelligence, there are similar theories, such as frame theory and script theory [20], [21].

Briars and Larkin proposed the CHIPS model in 1984 [22]. The idea of this model was to establish a problem schema while reading the sentences, and executed a series of actions according to the rules to solve the problem. When the system read words from the question text, CHIPS created elements in the working memory that represented the knowledge, such as chips, sets, set identifiers, time, members, and descriptors. Meantime, the system created a schema by storing the knowledge in a structured and organized way. For example, moving schemas recorded the knowledge about how the chips were moved from one set to another set and the conditions for stopping moving.

Riley et al. classified conceptual knowledge of math word problems into three categories: problem schemas for understanding semantic relationships, action schemas for representing knowledge of problem-solving actions, and strategy knowledge for solving problems [23]. When solving a problem, the problem schema was first used to represent the problem scenario, and then the action schema was used to generate the answer based on the defined strategy knowledge. The model attempted to model different levels of children abilities.

Kintsch and Greeno presented a model that combines Van Dijk and Kintsch's text understanding theory and Riley's assumption of understanding semantic knowledge of problems [24]. This model simulated the construction of cognitive representation in the process of solving the problem. It included the set of knowledge structures and the set of strategies to construct representation and solve problems. Fletcher implemented WORDPRO system with Interlisp-D language based on Kintsch's theory [25]. WORDPRO system constructed a two-level of representation: textbase and problem model. Textbase is an organized set of propositions, and the problem model is a non-propositional, domain-specific representation that drives the process of problem solving. WORDPRO system defined four modes: change-in, change-out, combine, and compare. It transformed the problem into propositional logic and used logical reasoning to obtain the final answer. ARITHPRO system is the third stage of the model that simulates Kintsch's problem-solving process [26]. The ROBUST system expanded change schema into six categories: Transfer-In-Ownership, Transfer-Out-Ownership, Transfer-In-Place, Transfer-Out-Place, Creation, and Termination, which could better express the change of owner or location [3]. Each change scheme corresponds to only one change formula.

Schema representation has a rich representation ability, but it also heavily relies on a set of human-crafted rules. It can only solve problems that satisfy the pre-defined rules, and cannot effectively solve unknown problems.

IV. STATISTICAL MACHINE LEARNING METHODS

Statistical machine learning methods can effectively overcome the disadvantage of rule-based methods by training models from labelled datasets. These methods extract features from input sequences or their syntax, and predict the output using support vector machines, log-linear models, etc. These models could use different types of problem representations. According to different representations, we classify the methods into transition-based methods, template-based methods, tree-based methods, and semantic representation methods.

A. *Transition-based Methods*

Transition-based methods predict the transition types of current problem states, and obtain the logical representation or equation of the problem. Different from rule-based methods, this method uses statistical machine learning methods to predict the verb category or problem type in the sentence [1], [27], [28]

Hosseini et al. proposed ARIS system, which viewed math word problem as a description of the problem states, and then used actions to describe the transition of the problem state [1]. ARIS system divided the question text into multiple sentence fragments, and each sentence described the observation or update of the current state. The description of the problem state included entities, containers, attributes, quantities, and relations. For the problem, the verb category of each sentence segment can be predicted based on support vector machine (SVM), and the state of the problem was constantly updated according to the verb category. Therefore, it generated a state representation of the entire problem and then established an equation representation. Hosseini created a dataset named AI2-395. The system can be used for addition and subtraction arithmetic word problems.

Sundaram proposed a system based on natural language processing techniques to extract problem information [29], and used a schema-based method in the inference phase. The system first used the Stanford CoreNLP tool to analyze the problem text and extract the information, and decomposed the problem into multiple clauses through clause identification, currency pre-processing, coreference resolution, and entity recognition. Then, the state of the problem was updated continuously based on pre-defined schema rules. Finally, the equations were created from the states of the whole problem.

B. *Template-based Methods*

Templates are abstract representations of equations or arithmetic formulas for math word problems. Template-based method first finds the template for the problem from a pre-defined template set, then identifies the entities and quantities in the question, fills them into the slots of the template, and finally calculates the answer. This method requires manual annotation to build a template set. Table 3 show a template of a given equation.

TABLE 3
A TEMPLATE EXAMPLE

Equation	$2*x-3*y-10=0$
Template	$v_1 \times u_1 - v_2 \times v_2 - n_3 = 0$

Kushman collected a dataset Alg-514 from Algebra.com, in which each problem was labeled with an equation template. Kushman solved the problem by finding the equation template for the problem and extracting information from the problem text to fill the slots in the template [5]. The optimization objective of the model was to maximize the log-likelihood of the candidates. The optimization objective of the model is shown in equation 1, where $V_i(y)$ is the indicator function to determine whether *y* is correct. The model is optimized using the L-BFGS method. If considering all possible quantities and templates, it would cause a very large search complexity in the derivation phase. The model adopted a beam search strategy to keep the best *k* candidate templates in the search process, and each template keeps the best *l* candidates.

$$O = \sum_i \sum_{\substack{y \in Y \\ s.t. V_i(y)=1}} p(y \mid x_i; \theta) \qquad (1)$$

Zhou also predicted the equation template to solve math word problem [30]. Different from Kushman's method, it filled the slots with the quantities extracted from the question, and designed effective features to describe the relations between quantities. This strategy greatly reduced the search space. In this method, max-margin method was used as the optimization objective of log-linear model.

Roy proposed a quantity reasoning method to solve the problem, including four steps: quantity extraction, quantity pair classification, operator classification and order classification [31]. Firstly, the model was used to identify the quantities of the problem, and their features included characters, word categories and part-of-speech tags [32]. Then, it obtained the quantity pairs and the operators through reasoning.

Mitra and Baral proposed a formula template representation method for addition and subtraction word problems. It described three types of problems: part-whole, change and compare [27]. This method first used a log-linear model to select the formula with the highest probability as the formula template, and then converted the formula into an arithmetic expression. Upadhyay proposed a weakly supervised learning method, which effectively used the data without equation annotation [33]. This method first trained an intermediate model using the data with equation annotation and result annotation, and then combined two kinds of data for mixed training.

Huang proposed a two-stage answering system [34]. The first stage was template retrieval, where ranking SVM method was used to obtain candidate equation templates that were most relevant to the problem, using the text and quantitative features. The second stage was alignment sorting, which sorted all alignments of the candidate equation templates, and found the best template and its alignment. Compared with the method that directly mapped the problem to the problem template, the two-stage method greatly reduced the search complexity.

Template-based methods are mainly based on pre-defined sets of templates, which are not scalable and cannot handle large datasets. It still needs to be extended to support more complex operations.

C. Tree-based Methods

Different from template-based method, tree-based methods directly predict tree-based arithmetic expressions or equations. Compared with flat structures, tree structures can more clearly express the hierarchical relations and operation priorities between quantities. For the problem with one unknown quantity, the equation tree has one more unknown variable node than the arithmetic tree. Figure 4 shows these two types of tree representation methods: arithmetic tree and equation tree. Both trees can be expressed as a binary tree structure, in which leaf nodes are known and unknown quantities, non-leaf nodes are operators. The operators with higher priority are at the low levels of the tree, and the root node of the tree is the operator with the lowest priority.

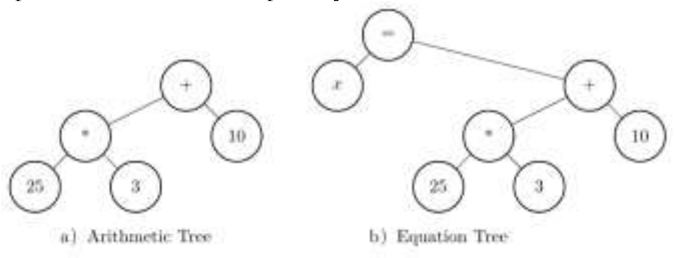

Fig. 1 Tree Representation

Roy proposed a bottom-up approach to convert arithmetic expressions into binary tree structures [35]. This method first trained a binary classifier to determine whether a quantifier was related to the final expression. Then, an expression tree was constructed through a series of decision-making processes, where each decision was used to determine the smallest common ancestor node between two quantifiers. Finally, the total score was calculated for the generated candidate trees, and the candidate tree with the highest score was selected as the final expression tree. The score of the candidate tree was shown in the formula 2, where IRR represented the probability of whether the quantity *q* was related to the final arithmetic formula, PAIR represented the probability that the quantifiers q_i and q_j choose T as the operator. To reduce the computational complexity, the method adopted the beam search algorithm, and kept the best *k* candidate trees in the search process. Roy simplified the decision-making process, but limits the scope of application [36].

$$\begin{aligned} Score(E) &= w_{IRR}\sum_{q\in I(E)} IRR(q) \\ &+ \sum_{q_i,q_j\in I(E)} PAIR(q_i,q_j,\odot_{LCA}(q_i,q_j,T)) \end{aligned} \tag{2}$$

Koncel-kedziorsk proposed ALGES system [6]. The system first extracted the quantities and their attributes to construct the set of quantities, then used operators to connect the quantities into a candidate equation tree, and finally used an integer linear programming method to obtain the final equation tree. The system differed from Roy's method in two ways [35]. One way was that it traversed all possible candidate trees over all quantities. The validity of candidate equation trees is determined by integer linear programming with constraints including syntactic validation, type consistency, and other constraints. The second difference was that it used all quantities as candidates.

Roy introduced unit dependency graph (UDG) to represent the dependencies between quantities [37]. In UDG, the vertices contained the extracted quantities, and the edges described the relations between two quantities. The probabilities of UDG were given in the equation below. The evaluation function was the sum of the probabilities of identifying the vertices and edges. This approach requires the annotation of UDG to train the classifier.

$$\begin{aligned} Score(G) &= \sum_{\substack{v\in V \\ Label(G,v)=RATE}} VERTEX(v,RATE) \\ &+ \sum_{v_i,v_j\in V,i<j} EDGE(v_i,v_j,LABEL(G,v_i,v_j)) \end{aligned} \tag{3}$$

Roy proposed declarative rules to transform a problem into mathematical expressions to understand mathematical concepts such as dimension and subset [7]. This method divided knowledge into two levels: mathematical concepts and declarative rules. Mathematical concepts were used for reasoning between quantities. Common mathematical concepts included transfer, proportion, part-whole relations.

The declarative rules were used to determine the operators. The probability of problem *x* and expression *y* is calculated by the sum of the probabilities of mathematical concepts $psi_k(x,k^o)$ and declarative rules $\psi_r(x,r^o)$, as shown in formula 4. This method mapped declarative knowledge into implicit variables.

$$Score(x,y)=\sum_{o\in\square(y)} w_k\psi_k(x,k^o)+w_r\psi_r(x,r^o) \quad (4)$$

Tree-based method overcomes the shortcomings of the template-based method. It only needs the arithmetic representation or equation representation of the problem. However, this method still needs to define the features such as quantitative features, quantitative pair features, verb features, and global features.

D. Semantic Representation Methods

Math word problems can also use other semantic representation methods. Shi adopts Dolphin Language (DOL) to represent math word problems [38]. Each question is represented as a DOL tree in which the nodes are divided into constants, categories, and functions. Constants include numbers, strings, and entities. Categories indicate the types of entities. Functions are used to combine small language units. This representation is based on context-free grammar. Finally, the best parse tree is selected and used to obtain the answer.

Huang and Liang proposed a tag-based model, which contains language analysis, problem parsing, and explanation generation models [39]–[41]. The language analysis module uses Stanford CoreNLP tool to convert the original question into a semantic representation tree, and each quantity is marked with grammatical and semantic information [42]. The problem analysis module contained three submodules. First, the SVM classifier was used to classify the type of the problem, and then the fact was extracted from the semantic representation tree. The first-order predicate logic was used to represent the problem, and finally the answer was obtained by the inference engine. The explanation generation module explains how to generate the answer to the question according to the derivation process. Liang proposed a meaning-based method [28], [43]. Each quantity is represented by one of the proposed role-tags (such as nsubj, verb, etc.). In logical conversion stage, the method introduces the SVM model to choose the quantity and the operator.

V. DEEP LEARNING METHODS

Deep learning methods learn effective feature representations directly from the data [44], and have been successful in many tasks such as machine translation, reading comprehension. For math word questions, one of the common methods is encoder-decoder models, including sequence-to-sequence models, attention mechanisms, and transformer [45]–[47]. These models encode the problem, and decode it into its representations, such as sequences of actions, mathematical equations, mathematical operators, and semantic representations.

E. Transition-based Methods

Ling regarded math word problems as finding the instruction sequence to the answer [48]. This method viewed each instruction as a tuple, including an operation, ordered sequence of parameters, result storage location, and operation parameters. The model used a sequence-to-sequence model to predict instruction sequence and introduced attention mechanism and copy mechanism to predict parameter sequence.

Chiang regarded the formula construction as the operation of a stack. It identified a quantity as pushing it into the stack, and calculated two quantities as operating the top two elements of the stack [49]. In this paper, an encoder-decoder framework was proposed to automatically solve math word problems by mapping the semantic representation of the problem to stack operation. In the model, the encoder was used to understand the semantics of the problem, and the decoder focused on deciding which symbol to generate.

Amini introduced a math word problem dataset MathQA. In the dataset, each problem is marked with operation steps and parameters (i.e. operands) [50]. The dataset defines 58 different operations. This paper adopted an encoder-decoder model based on LSTM and attention mechanism to map problems into sequences of operations. The experimental results on MathQA and AQuA data sets verified the effeciency of the method.

F. Template-based Methods

Mehta proposed DILTON system to solve math word problems [51]. The system applied a deep neural network model to predict the operator ('-','+','*','/'), and then used it to generate answers. The DILTON system divides the question into two parts: question status and question query. Question status and question query are processed separately by two different networks, and finally merged to predict the final operator.

Wang applied deep neural networks to math word problems. This method used a sequence-to-sequence model to map the problem to a mathematical equation [52]. First, all the words in the problem were represented as vectors by the word vector layer. In the encoding layer, GRU networks were used to construct the dependency relations between words and represent the sentence as a vector. The decoding layer converted the vector into an equation.

Huang et al. introduced the copy mechanism and alignment mechanism into the sequence-to-sequence model [53]. The copy mechanism directly coped the quantities into the equation, and the alignment mechanism was used to align the quantities in the problem with the quantities in the equation to avoid generating them in the wrong position. The output of the deep learning method was used as the input of Huang's two-stage model to achieve a combination of these two methods [34] .

Robaidek adopted the BiLSTM model and structured self-attention model in sequence-to-sequence model to improve the prediction accuracy [10]. Wang compared the performance of the BiLSTM model, the ConvS2S model and the transformer model, integrated the output probabilities of these three models, and choosed the output with the largest probability product [54].

Li proposed a group attention method inspired by the multi-head attention mechanism, including different attention mechanisms [11]. These attention mechanisms were used to extract different types of information, including global attention that described global information, quantity attention that represented the relationship between current quantity and its neighboring quantities, quantity pair attention that described the relations between the quantities, and problem attention that represented the relations between the problem and the quantity. Finally, these attentions are concatenated as group attention.

Wang proposed new type of templates that represented the questions with suffix expressions [55]. The attention mechanism was introduced to capture contextual semantic information. This method designed a recurrent neural network to infer the operators.

G. Tree-based Methods

Xie et al. used a tree-based neural network model to generate arithmetic trees [56]. Given a math word problem, an encoder-decoder model was used. The encoder model used a recurrent neural network based on GRU to encode the problem. The decoder used a top-down decoding approach with a depth-first strategy, starting from the head node of the tree. The node was first predicted by the attention model, whether it was a known quantity, an unknown quantity, or an operator. If the node was a quantity, it terminated the prediction and set the node as a leaf node. If the node was an operator, a two-layer feedforward neural network was used to predict the left and right subtrees of the operator, respectively.

Wang designed a framework by applying deep reinforcement learning methods to construct arithmetic trees [57]. They designed the states, actions, reward functions, and network structures. The model used a two-layer feedforward neural network to calculate the predicted Q value and minimizes the difference between the predicted Q value and the target Q value as a loss function. Zhang proposed an encoder-decoder model, where the encoder constructed two graphs related to quantities, and then encode the two graphs through graph convolutional networks and graph transformer, and the decoder adopted a tree-based decoder [58].

H. Semantic Representation Methods

Huang proposed an intermediate representation to reduce the difference between the problem and the formula [59]. Since the intermediate representation is unknown, the method introduced an iterative labelling framework. This method first derived possible intermediate representations and used these representations to pre-train a model. Then the intermediate representation was improved by the iteration of predicting the formula. The method used a sequence-to-sequence model with attention normalization terms to learn this intermediate representation and obtained the result by calculating the intermediate representation.

VI. GENERAL CHALLENGE AND FUTURE DIRECTIONS

Due to the difficulty of automatically solving elementary math word problems, the current research using deep learning methods is still in development. Compared with machine translation and question answering systems, it is still facing large challenge.

The current problems mainly include the following points:

(1) Semantic gap between the problem and the equation

At present, most methods express math word problems as equations or arithmetic formulas. It is difficult to directly learn the mapping relations between them. Compared with semantic parsing, there is little research on the representation of math word problems. Only Huang defines the DOL language as the language to represent the problems [60], and proposed an iterative method to find an intermediate semantic representation between the problem and the DOL language [59]. Therefore, it is worth finding appropriate semantic representation.

(2) The problem of small dataset size

Compared with the datasets of image processing and natural language processing problems, the size of math word problem datasets is normally small. The English dataset Dolphin18K contains 18,000 samples, and the Chinese dataset Math23K has 23,000 samples [52], [60]. It is another challenge that learns a deep neural network model on a small dataset.

(3) Lacking common-sense knowledge

Common-sense knowledge plays an important role in many natural language processing tasks, such as automatic question answering and recommendation systems. For math word problems, understanding the knowledge in the problem is also very important, but there is no further work on this topic.

VII. CONCLUSION

This paper briefly describes the methods of solving elementary math word problems. We introduce the rule-based methods, statistical machine learning methods, and deep learning methods. For each type of methods, we present different types of problem representation, such as transition-based, template-based, tree-based, and semantic representation methods. The characteristics and differences of these methods are compared and analyzed. Though it makes an improvement with the development of deep learning, there are still many problems that need to solve in the future.

REFERENCES

[1] M. J. Hosseini, H. Hajishirzi, O. Etzioni, and N. Kushman, “Learning to Solve Arithmetic Word Problems with Verb Categorization,” in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, Oct. 2014,

pp. 523–533. [Online]. Available: http://www.aclweb.org/anthology/D14-1058

[2] D. Zhang, L. Wang, L. Zhang, B. T. Dai, and H. T. Shen, "The Gap of Semantic Parsing: A Survey on Automatic Math Word Problem Solvers," IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–1, 2019, doi: 10.1109/TPAMI.2019.2914054.

[3] Y. Bakman, "Robust Understanding of Word Problems with Extraneous Information," arXiv:math/0701393, Jan. 2007, Accessed: May 04, 2019. [Online]. Available: http://arxiv.org/abs/math/0701393

[4] A. Mukherjee and U. Garain, "A Review of Methods for Automatic Understanding of Natural Language Mathematical Problems," Artificial Intelligence Review, vol. 29, no. 2, pp. 93–122, Apr. 2008, doi: 10.1007/s10462-009-9110-0.

[5] N. Kushman, Y. Artzi, L. Zettlemoyer, and R. Barzilay, "Learning to Automatically Solve Algebra Word Problems," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, Maryland, Jun. 2014, pp. 271–281. doi: 10.3115/v1/P14-1026.

[6] R. Koncel-Kedziorski, H. Hajishirzi, A. Sabharwal, O. Etzioni, and S. D. Ang, "Parsing Algebraic Word Problems into Equations," Transactions of the Association for Computational Linguistics, vol. 3, pp. 585–597, Dec. 2015, doi: 10.1162/tacl_a_00160.

[7] S. Roy and D. Roth, "Mapping to Declarative Knowledge for Word Problem Solving," Transactions of the Association for Computational Linguistics, vol. 6, pp. 159–172, Jul. 2018, doi: 10.1162/tacl_a_00012.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv:1810.04805 [cs], Oct. 2018, Accessed: May 04, 2019. [Online]. Available: http://arxiv.org/abs/1810.04805

[10] B. Robaidek, R. Koncel-Kedziorski, and H. Hajishirzi, "Data-Driven Methods for Solving Algebra Word Problems," CoRR, vol. abs/1804.10718, 2018, [Online]. Available: http://arxiv.org/abs/1804.10718

[11] J. Li, L. Wang, J. Zhang, Y. Wang, B. T. Dai, and D. Zhang, "Modeling Intra-Relation in Math Word Problems with Different Functional Multi-Head Attentions," in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, Jul. 2019, pp. 6162–6167. doi: 10.18653/v1/P19-1619.

[12] S. Mandal and S. K. Naskar, "Solving arithmetic mathematical word problems: A review and recent advancements," in Information technology and applied mathematics, Singapore, 2019, pp. 95–114.

[13] D. G. Bobrow, "Natural Language Input for a Computer Problem Solving System," Department of Mathematics, MIT, Cambridge, 1964. Accessed: Apr. 16, 2019. [Online]. Available: http://dspace.mit.edu/handle/1721.1/6903

[14] J. R. Slagle, "Experiments with a Deductive Question-answering Program," Communications of the ACM, vol. 8, no. 12, pp. 792–798, Dec. 1965, doi: 10.1145/365691.365960.

[15] E. Charniak, "CARPS: A Program which Solves Calculus Word Problems," MIT, Cambridge, 1968. Accessed: Apr. 16, 2019. [Online]. Available: http://dspace.mit.edu/handle/1721.1/6901

[16] J. P. Gelb, "Experiments with a Natural Language Problem-Solving System," in Proceedings of the 2nd International Joint Conference on Artificial Intelligence., London, UK, Sep. 1971, pp. 455–462. Accessed: Apr. 19, 2019. [Online]. Available: http://ijcai.org/Proceedings/71/Papers/041.pdf

[17] R. F. Simmons, "Natural Language Question-answering Systems: 1969," Commun. ACM, vol. 13, no. 1, pp. 15–30, Jan. 1970, doi: 10.1145/361953.361963.

[18] E. Charniak, "Computer Solution of Calculus Word Problems," in Proceedings of the 1st International Joint Conference on Artificial Intelligence, Washington, DC, USA, May 1969, pp. 303–316. [Online]. Available: http://ijcai.org/Proceedings/69/Papers/031.pdf

[19] R. C. Anderson, "The Notion of Schemata and the Educational Enterprise: General Discussion of the Conference," in Schooling and the Acquisition of Knowledge, R. C. Anderson, R. J. Spiro, and W. E. Montague, Eds. Lawrence Erlbaum, 1984, pp. 415–31.

[20] M. Minsky, "A Framework for Representing Knowledge," in The Psychology of Computer Vision, New York: McGraw-Hill, 1975, pp. 211–277.

[21] R. C. Schank and R. P. Abelson, Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures. Hillsdale, NJ: L. Erlbaum, 1977.

[22] D. J. Briars and J. H. Larkin, "An Integrated Model of Skill in Solving Elementary Word Problems," Cognition and Instruction, vol. 1, no. 3, pp. 245–296, 1984.

[23] M. S. Riley, J. G. Greeno, and J. I. Heller, "Development of Children's Problem-Solving Ability in Arithmetic," in The Development of Mathematical Thinking, Orlando, FL: Academic Press, Inc., 1984. Accessed: Apr. 16, 2019. [Online]. Available: https://eric.ed.gov/?id=ED252410

[24] W. Kintsch and J. G. Greeno, "Understanding and Solving Word Arithmetic Problems," Psychological Review, vol. 92, no. 1, pp. 109–129, Jan. 1985.

[25] C. R. Fletcher, "Understanding and Solving Arithmetic Word Problems: A Computer Simulation," Behavior Research Methods, Instruments, & Computers, vol. 17, no. 5, pp. 565–571, Sep. 1985, doi: 10.3758/BF03207654.

[26] D. Dellarosa, "A Computer Simulation of Children's Arithmetic Word-Problem Solving," Behavior Research Methods, Instruments, & Computers, vol. 18, no. 2, pp. 147–154, Mar. 1986, doi: 10.3758/BF03201014.

[27] A. Mitra and C. Baral, "Learning To Use Formulas To Solve Simple Arithmetic Problems," in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, Aug. 2016, pp. 2144–2153. doi: 10.18653/v1/P16-1202.

[28] C.-C. Liang, Y.-S. Wong, Y.-C. Lin, and K.-Y. Su, "A Meaning-Based Statistical English Math Word Problem Solver," in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, Louisiana, Jun. 2018, pp. 652–662. doi: 10.18653/v1/N18-1060.

[29] S. S. Sundaram and D. Khemani, "Natural Language Processing for Solving Simple Word Problems," in Proceedings of the 12th International Conference on Natural Language Processing, Trivandrum, India, Dec. 2015, pp. 394–402. [Online]. Available: https://www.aclweb.org/anthology/W15-5955

[30] L. Zhou, S. Dai, and L. Chen, "Learn to Solve Algebra Word Problems Using Quadratic Programming," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, Sep. 2015, pp. 817–822. doi: 10.18653/v1/D15-1096.

[31] S. Roy, T. Vieira, and D. Roth, "Reasoning about Quantities in Natural Language," Transactions of the Association for Computational Linguistics, vol. 3, pp. 1–13, Dec. 2015, doi: 10.1162/tacl_a_00118.

[32] J. D. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in Proceedings of the Eighteenth International Conference on Machine Learning, San Francisco, CA, USA, 2001, pp. 282–289.

[33] S. Upadhyay, M.-W. Chang, K.-W. Chang, and W. Yih, "Learning from Explicit and Implicit Supervision Jointly for Algebra Word Problems," in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, Nov. 2016, pp. 297–306. doi: 10.18653/v1/D16-1029.

[34] D. Huang, S. Shi, C.-Y. Lin, and J. Yin, "Learning Fine-Grained Expressions to Solve Math Word Problems," in Proceedings of the 2017 conference on empirical methods in natural language processing, Copenhagen, Denmark, Sep. 2017, pp. 805–814. doi: 10.18653/v1/D17-1084.

[35] S. Roy and D. Roth, "Solving General Arithmetic Word Problems," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, Sep. 2015, pp. 1743–1752. doi: 10.18653/v1/D15-1202.

[36] S. Roy, S. Upadhyay, and D. Roth, "Equation Parsing : Mapping Sentences to Grounded Equations," in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, Nov. 2016, pp. 1088–1097. doi: 10.18653/v1/D16-1117.

[37] S. Roy and D. Roth, "Unit Dependency Graph and Its Application to Arithmetic Word Problem Solving," in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA, 2017, pp. 3082–3088. Accessed: May 04, 2019. [Online]. Available: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14764

[38] S. Shi, Y. Wang, C.-Y. Lin, X. Liu, and Y. Rui, "Automatically Solving Number Word Problems by Semantic Parsing and Reasoning," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, Sep. 2015, pp. 1132–1142. doi: 10.18653/v1/D15-1135.

[39] C.-T. Huang et al., "Designing a tag-based statistical math word problem solver with reasoning and explanation," in Proceedings of the 27th conference on computational linguistics and speech processing (ROCLING 2015), Hsinchu, Taiwan, Oct. 2015, pp. 58–63. [Online]. Available: https://www.aclweb.org/anthology/O15-1006

[40] C.-T. Huang, Y.-C. Lin, and K.-Y. Su, "Explanation Generation for a Math Word Problem Solver," in Proceedings of the 27th Conference on Computational Linguistics and Speech Processing (ROCLING 2015), Hsinchu, Taiwan, Oct. 2015, pp. 64–70. [Online]. Available: https://www.aclweb.org/anthology/O15-1007

[41] C.-C. Liang, K.-Y. Hsu, C.-T. Huang, C.-M. Li, S.-Y. Miao, and K.-Y. Su, "A Tag-based English Math Word Problem Solver with Understanding, Reasoning and Explanation," in Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, San Diego, California, Jun. 2016, pp. 67–71. doi: 10.18653/v1/N16-3014.

[42] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," in Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, Maryland, Jun. 2014, pp. 55–60. doi: 10.3115/v1/P14-5010.

[43] C.-C. Liang, S.-H. Tsai, T.-Y. Chang, Y.-C. Lin, and K.-Y. Su, "A Meaning-Based English Math Word Problem Solver with Understanding, Reasoning and Explanation," in Proceedings of COLING 2016, the 26th international conference on computational linguistics: system demonstrations, Osaka, Japan, Dec. 2016, pp. 151–155.

[44] L. Dong, F. Wei, M. Zhou, and K. Xu, "Question Answering over Freebase with Multi-Column Convolutional Neural Networks," in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint

Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, Jul. 2015, pp. 260–269. doi: 10.3115/v1/P15-1026.

[45] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in Advances in neural information processing systems, 2014, vol. 27. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf

[46] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in 3rd international conference on learning representations, ICLR 2015, san diego, CA, USA, may 7-9, 2015, conference track proceedings, 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[47] A. Vaswani et al., "Attention is all you need," in Advances in neural information processing systems 30: Annual conference on neural information processing systems 2017, long beach, CA, USA, Dec. 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need

[48] W. Ling, D. Yogatama, C. Dyer, and P. Blunsom, "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), Vancouver, Canada, Jul. 2017, pp. 158–167. doi: 10.18653/v1/P17-1015.

[49] T.-R. Chiang and Y.-N. Chen, "Semantically-Aligned Equation Generation for Solving and Reasoning Math Word Problems," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, Jun. 2019, pp. 2656–2668. doi: 10.18653/v1/N19-1272.

[50] A. Amini, S. Gabriel, S. Lin, R. Koncel-Kedziorski, Y. Choi, and H. Hajishirzi, "MathQA: Towards Interpretable Math Word Problem Solving with Operation-Based Formalisms," in Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), Minneapolis, Minnesota, Jun. 2019, pp. 2357–2367. doi: 10.18653/v1/N19-1245.

[51] P. Mehta, P. Mishra, V. Athavale, M. Shrivastava, and D. Sharma, "Deep Neural Network Based System for Solving Arithmetic Word Problems," in Proceedings of the IJCNLP 2017, system demonstrations, Tapei, Taiwan, Nov. 2017, pp. 65–68.

[52] Y. Wang, X. Liu, and S. Shi, "Deep Neural Solver for Math Word Problems," in Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, Sep. 2017, pp. 845–854. doi: 10.18653/v1/D17-1088.

[53] D. Huang, J. Liu, C.-Y. Lin, and J. Yin, "Neural Math Word Problem Solver with Reinforcement Learning," in Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, New Mexico, USA, Aug. 2018, pp. 213–223.

[54] L. Wang, Y. Wang, D. Cai, D. Zhang, and X. Liu, "Translating a Math Word Problem to a Expression Tree," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, Oct. 2018, pp. 1064–1069. [Online]. Available: https://www.aclweb.org/anthology/D18-1132

[55] L. Wang et al., "Template-Based Math Word Problem Solvers with Recursive Neural Networks," in The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, Feb. 2019, pp. 7144–7151. doi: 10.1609/aaai.v33i01.33017144.

[56] Z. Xie and S. Sun, "A Goal-Driven Tree-Structured Neural Model for Math Word Problems," in Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19, Jul. 2019, pp. 5299–5305. doi: 10.24963/ijcai.2019/736.

[57] J. Zhang et al., "Graph-to-tree learning for solving math word problems," in Proceedings of the 58th annual meeting of the association for computational linguistics, Online, Jul. 2020, pp. 3928–3937. doi: 10.18653/v1/2020.acl-main.362.

[58] Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. MathDQN: Solving Arithmetic Word Problems via Deep Reinforcement Learning. In Proceedings of the Thirty-Second {AAAI} Conference on Artificial Intelligence, (AAAI-18), pages 5545–5552, New Orleans, Louisiana, USA, February 2018.

[59] D. Huang, J.-G. Yao, C.-Y. Lin, Q. Zhou, and J. Yin, "Using Intermediate Representations to Solve Math Word Problems," in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, Jul. 2018, pp. 419–428.

[60] D. Huang, S. Shi, C.-Y. Lin, J. Yin, and W.-Y. Ma, "How well do Computers Solve Math Word Problems? Large-Scale Dataset Construction and Evaluation," in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, Aug. 2016, pp. 887–896.