International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Solving Arithmetic Word Problems Using Natural Language Processing and Rule-Based Classification

Swagata Acharya [1], Sourav Mandal [*2], Rohini Basak [3]

Submitted: 28/09/2021 Accepted : 02/02/2022

Abstract: In today's world, intelligent tutoring systems (ITS), computer-based training (CBT), etc. are rapidly gaining popularity in both educational and professional fields, and an automatic solver for mathematical word problems is one of the most important subfields of ITS. Automatic solving of mathematical word problems is a challenging research problem in the fields of artificial intelligence (AI) and its subfields like natural language processing (NLP), machine learning (ML), etc., since understanding and extracting relevant information from an unstructured text requires a lot of logical skills. To date, much research has been done in this area, focusing on solving each type of mathematical word problem, such as arithmetic word problems, algebraic word problems, geometric word problems, trigonometric word problems, etc. In this paper, we present an approach to automatically solve arithmetic word problems. We use a rule-based approach to classify word problems. We propose various rules to establish the relationships and dependencies among different key elements and classify the word problems into four categories (Change, Combine, Compare, and Division-Multiplication) and their subcategories to identify the desired operation among+, -, *, and /. However, it is limited to solving only word problems with a single operation and a single equation word problem. Irrelevant information is also filtered out from the input problem texts, based on manually created rules to extract relevant quantities. Later, an equation is formed with the relevant quantities and the predicted operation to obtain the final answer. The work proposed here performs well compared to most similar systems based on the standard SingleOp dataset, achieving an accuracy of 93.02%.

Keywords: *solving* arithmetic *word problems, classification of word problems, rule-based information extraction, rule-based arithmetic word problem solver.*

This is an open access article under the CC BY-SA 4.0 license.
(https://creativecommons.org/licenses/by-sa/4.0/)

1. Introduction

Mathematical word problems can be defined as mathematical exercises that present the relevant background information on a problem as natural language text, rather than in the form of mathematical notations [1]. Here, the natural language could be any language like English, Arabic etc. For example, “Kimberly has 5 Skittles. She buys 7 more. Later, Kimberly buys 18 oranges at the store. How many Skittles does Kimberly have in all?” is an arithmetic word problem from our reference dataset. Where, as per the question asked, 5 and 7 are treated as relevant quantities and take part to generate the final answer (5+7) =12, and 18 is considered as irrelevant quantity. Since, the arithmetic word problems are an integral part of our day-to-day calculation, even the children get introduced to such kind of problems from a very early stage and as they move to higher classes, the complexity of the problems also get increased. The children are trained to solve word problems involving the basic mathematical operations like, addition, subtraction, multiplication, and division in beginning, to rate, permutation, combination, probability etc.
However, though the human brain can solve all these kinds of diverse problems efficiently along with their growing age and learning experiences, it is quite difficult for a system to solve even a basic addition-subtraction kind of word problem. Though, from the evaluation of the computer, the machine has proven its supremacy in terms of speed and accuracy of the mathematical calculations, it is not capable enough to extract information from natural languages and understand accordingly. Therefore, it is considered as one of the open research areas in the domain AI, ML, NLP, to build a computing system which can be programmed according to human cognitive perspective to solve mathematical word problems in different ways.
Designing the algorithms to solve mathematical word problems is not a new concept, the idea arose back in the 1960s and since then plenty of research have been carried out to deal with various aspects of solving diverse word problems. Although, the progress is not that great and still in the preliminary level. We have chosen to work on SingleOp dataset, which was published by [2], containing arithmetic word problems with all the basic four operations (+, -, *, /).
The Intelligent Tutoring Systems (ITS), Computer Based Training (CBT), and various online-learning platforms are also gaining popularity in the last two decades. These systems are trying to use AI, ML, and NLP to improve the quality of teaching-learning procedure. Mathematical word problem solvers can be one useful components of such systems, which typically focus to replicate the personalized tutoring. The authors in [3-6], have made some attempts to build intelligent tutoring systems related to arithmetic word problems. The primary goal of these

[1] *Dept. of Information Technology, Jadavpur University, Kolkata- 700032, West Bengal, India. ORCID ID: 0000-0002-4942-5734*
[2] *School of Computer Science and Engineering (SCSE), XIM University, Harirajpur-752050, Odisha, India. ORCID ID: 0000-0002-6066-8008*
[3] *Dept. of Information Technology, Jadavpur University, Kolkata- 700032, West Bengal, India. ORCID ID: 0000-0001-9662-3074*
** Corresponding Author Email: Sourav_officials@yahoo.co.in*

Table 1. Example of MWPS belongs to various classes and sub-classes.

Change (Level 1)	
Change Plus (Level 2) There were 14 kids on the soccer field. 22 kids decided to join in. Now how many kids are on the soccer field? Relevant quantities- 14, 22, Operation- '+', Equation- 14+22, Answer- 36	Change Minus (Level 2) Denise removes 5 bananas from a jar. There were originally 46 bananas in the jar. How many bananas are left in the jar? Relevant quantities- 5, 46, Operation- '-', Equation- 46-5, Answer- 41
Combine (Level 1)	
Combine Plus (Level 2) A cake recipe requires 0.6 cup of sugar for the frosting and 0.2 cup of sugar for the cake. How much sugar is that altogether? Relevant quantities- 0.6, 0.2, Operation- '-', Equation- 0.6+0.2, Answer- 0.8	Combine Minus (Level 2) There are 40 boys and some girls on the playground. There are 117 children altogether. How many girls are on the playground? Relevant quantities- 40, 117, Operation- '-', Equation- 117-40, Answer- 77
Compare (Level 1)	
Compare Plus (Level 2) Lucy has an aquarium with 212 fish. She wants to buy 68 more fish. How many fish would Lucy have then? Relevant quantities- 212, 68, Operation- '+', Equation- 212+68, Answer- 280	Compare Minus (Level 2) James has 232 balloons. Amy has 101 balloons. How many more balloons does James have than Amy? Relevant quantities- 232, 101, Operation- '-', Equation- 232-101, Answer- 131
Division-Multiplication (Level 1)	
Division (Level 2) Betty has 24 oranges stored in boxes. If there are 3 boxes, how many oranges must go in each box? Relevant quantities- 24, 3, Operation- '/', Equation- 24/3, Answer- 8	Multiplication (Level 2) Jill invited 37 people to her birthday party. They each ate 8 pieces of pizza. How many pieces of pizza did they eat? Relevant quantities- 37, 8, Operation- '*', Equation- 37*8, Answer- 296

systems is to provide high quality education to each student through computers.

We have tried to build an arithmetic word problem solver by classifying the word problems according to their operations involved and our work is inspired by the research of [7-10]. After classifying the word problems, we solved them independently. Table 1 shows some word problems belonging to various categories and sub-categories.

Depending on various research studies, the authors in [7], tried to categorize the addition-subtraction type arithmetic word problems into four categories i.e., CHANGE, EQUALIZING, COMBINE and COMPARE. Further, by performing deeper analysis, authors in [7] tried to sub-categorize them. The sub-categories are, CHANGE (Result unknown, change unknown, start unknown), EQUALIZING, COMBINE (Combine value unknown, Subset unknown), COMPARE (Difference unknown, Compared quantity unknown, Referent unknown) [7, Table 4.3]. Among these, the categories CHANGE, COMBINE and COMPARE are also used by the authors in [11-13], though the sub-category names differ.

The authors in [8, 9], also perform similar kind of classification techniques with CHANGE, COMBINE, and COMPARE [8, Table 1] or with the names differ slightly [9, Table 2].

The authors in [14], first proposed a method, which includes the division-multiplication type problems along with addition-subtraction type problems. All the systems proposed by [2, 10, 14, 15, 16] can solve the arithmetic problems with all the basic four types of operations. The datasets chosen by them are also similar. All this research motivated us to solve the addition, subtraction, multiplication, and division type word problems with some new strategies.

The core concept of our technical approach is based on classification features we proposed, which is also closely related to the human cognitive capabilities in understanding natural language-based word problems. We used several keyword-based cues to classify the word problems into four categories i.e., Change, Combine, Compare and Division-Multiplication. This concept is quite like the work of [10]. Further we sub-categorised these categories to identify the desired operation, using keyword-based cues, pattern-based cues, parts-of-speech cues etc. We also worked with the problems containing irrelevant quantities and used rule-based approaches to identify relevant quantities from the word problem discussed later in this paper. Defining rules for the problems which require numerical reasoning or word knowledge was quite difficult for our approach, though we tried to relate those problems to some structural cues, most of the time it does not satisfy the proper reasoning. The source code of our work is available at [43]. The key highlights of this research work are given below.

- Innovative classification features extraction.
- Unique rules for irrelevant information removal.

The rest of the paper is organized as follows. Section 2 describes the previous work done in the field of automatic math word problem solving. Then, the detailed methodology we followed is described in section 3. Section 4 discusses the experimental results critically and analyse the errors of our proposed method. Finally, section 5 concludes the paper and provides with the future scope of the proposed method.

2. Related Work

We can broadly categorize the methodologies adopted in previous work into three categories- using Symbolic Semantic Parsing, using Structure Prediction, and using Deep Learning. In Symbolic Semantic Parsing [17,18, 19, 20, 21, 22], semantic parsing refers to the process of converting natural language text (parsing) to an intermediate logical form that captures the meaning of the input (semantic). We refer to the early work in this field as symbolic semantic parsing as the intermediate representation often including human readable symbols. Structure Prediction [2, 9, 15, 16, 23, 24, 25, 26, 27, 28, 29], refers to the process of building data driven models that align a particular simple intermediate representation with the vectorial representation of a word problem. The features used to convert the text to vector were hand-engineered. Deep Learning [30] modelling is a recent neural network-based approach to convert one sequence to another.

[17] first proposed an approach and built a system named "STUDENT", which is capable to read, understand and evaluate a wide range of algebraic word problems (of some specific structure like, times, rates, percentage etc.,) represented in natural language (English) and gives the answer in natural language. The system basically consists of two programs- (i) "STUDENT" is for converting the algebraic word problems into equation form and (ii) "REMEMBER" is for storing the global information for solving a particular word problem. However, the main disadvantage of the system is, it is only able to solve very small number of problems due to the limited semantic base and global information. After, the work of [17], the next notable work was done by [31], and they discussed about the issues, that children face during solving an arithmetic word problem. One of these issues is about establishing the relationship between conceptual and procedural knowledge. By analysing the characteristics of the addition-subtraction type word problems, they proposed a theoretical approach to solve the problems by categorizing them into four categories- 'CHANGE', 'COMBINE', 'COMPARE' and 'EQUALIZING'. These categories are further divided into sub-categories. The categories, 'CHANGE' and 'EQUALIZING' are generally associated with the actions, related to increment or decrement of the quantities, but 'COMBINE' and 'COMPARE' describe the static relationship among the quantities.

Another, similar kind of work was done by [32]. They built a model named 'CHIPS', which was a simulation program based on children psychology for solving any word problem and the level of difficulties they met. The similar kind of work was also carried by [33] and [8], and the system they built is based on human cognitive science theories, named 'ARITHPRO' and 'WORDPRO' respectively. To represent the meaning of a word problem, 'WORDPRO' uses a set a proposition. The system consists of four schemas- 'Change-in', 'Change-out', 'Combine' and 'Compare'. Conceptually, the 'Change' schema, establish the relationship among, 'Start-set' ('Jemmy had 5 apples'), 'Transfer-set' ('Then, John gave her 4 apples') and 'Result-set'('How many apples does Jemmy have now?'). Basically, the system solves a problem by following certain rules, i.e., 13 rules for 'meaning postulates', 12 rules for 'arithmetic strategies' and 11 rules for 'problem solving procedures'. These rules are applied sequentially, for addition, change and subtraction, depending on the content of the 'Short Term Memory'(STM) of the system. Just like 'CHIPS' and 'WORDPRO', 'ARITHPRO' could also solve the single equation and single operation word problems of type addition-subtraction. It also categorizes the word problems into three categories ('CHANGE', 'COMBINE', 'COMPARE'), same as 'CHIPS'. Since, both systems had limitations on the change verb ('give') and the order of appearance of the problem sentences i.e., the first sentence must mention the number of objects the owner had initially, and the second sentence must contain the change verb. After, the previous research, the next remarkable research was done by [19], with the system 'ROBUST', which could understand free-format multi-step arithmetic word problems with irrelevant information. Though, the system is based on propositional logic, it can work perfectly for multiple verbs and the corresponding operations. Instead of identifying the operation, 'ROBUST' uses the concept of schema. The author expanded the 'CHANGE' schema into six distinct categories ('Transfer-In-Ownership', 'Transfer-Out-Ownership', 'Transfer-In-Place', 'Transfer-Out-Place', 'Creation' and 'Termination') according to their role in the word problem. The author in [21], proposed an approach and came out with the system 'ARIS', which solves addition-subtraction type problems by following Symbolic Semantic Parsing. It represents the whole word problem as a logic template named state, which consists of a set of entities, their attributes, containers, quantities, and their relationships. For example, "Tom has 10 white kittens.", here, 'kitten' refers to the entity, 'white' refers to its attribute, and since kittens belongs to Tom, so, 'Tom' is treated as the container. They built an SVM-based classifier to identify the verb category. They also compiled a dataset named AI [36] on addition-subtraction type problems and achieved remarkable accuracy on that dataset.

Some notable works on tree-based methods (structure prediction) have been done by [2,23,24]. The main idea behind their works is to transform the arithmetic expression to an equivalent binary tree structure step-by-step, by following bottom-up approach, where the internal nodes represent the operators, and the leaves represent the operands. The main advantage of this is, there is no need of additional annotations i.e., equation template, logic forms or tags. The algorithmic approach they developed, can solve multi-step and multi-operation arithmetic word problems and the algorithmic framework consists of two processing stages. At the first stage, the relevant quantity extraction is done from the input text and from the bottom levels of the tree. The syntactically valid candidate trees but, with different internal nodes and structures are enumerated. At the second stage, to pick the best candidate tree, they defined a scoring function, and that candidate tree is used to derive the final output. All the algorithms they developed, follows a common strategy to build the local classifier to predict the operation between two quantities. The authors in [2], first proposed the algorithmic approach of expression tree to solve the arithmetic word problems. They trained a binary classifier to determine whether the extracted quantity is relevant or not, to minimize the search space. Only the relevant quantities take part in tree construction, and are placed at the bottom level, while the irrelevant quantities are eliminated. They introduced and proved many theorems to identify the operations between two relevant quantities along with their order of occurrence. They used multiclass SVM to predict the operation and the binary SVM to identify relevant quantities. They out-performed all the previous system accuracies on existing datasets and created two new datasets, named Illinois-562 and Commoncore-600, consisting of more diverse and complex word problems. Their system was more generalized with minimal dataset dependency. Further, they extended their work to create a web based MWP solver [23], which can solve a huge number of word problems provided by common users. To manage the queries, asking for operations between the numbers, they added a CFG parser with their existing MWP solver. Later, they also developed a system based on the theory of 'Unit Dependency Graph' (UDG), which identifies the relationship and dependency between the units of the quantities [24]. An extensive review of these works can be found in [37].

The authors in [15], first approach the method based on template-based techniques (structure prediction) to solve the algebraic word problems. The area of their research was based on three main fields (Semantic Interpretation, Information Extraction, and Automatic Word Problem Solver) of Natural Language Processing (NLP). They used both supervised and semi-supervised learning methods by gathering problems and solutions from a website named Algebra.com. However, the performance of their system was not up to the mark, where the additional background knowledge and domain knowledge were required. For example, "A painting is 20 inches tall and 25 inches wide. A print of the painting is 35 inches tall, how wide is the print in inches?" However, they included all the basic four operations (+, -, *, /) in their dataset. In Equation Based, many of the systems can handle multiple simultaneous equations. However, if the template is not available in the training phase, it is not possible to generate new templates at inference time. In structure prediction, every system had their own niche set of hand-crafted rules to develop the vector representation of a word problem. They were far more generalizable than their symbolic counterparts. There were some attempts at modelling domain knowledge either in the form of constraints or introducing new template elements [38].

In the recent years, deep learning has gained remarkable popularity due to its predominance in terms of accuracy, when, the system is trained with enough data. In the last few years, several efforts have been put into solving math word problems by applying deep learning. The authors in [30], first proposed an algorithm Deep Neural Solver (DNS) which does not depend on hand-crafted features, and it is considered as huge contribution, since it does not require any human intelligence for feature extraction. It directly translates the input word problem to corresponding equation templates using Recurrent Neural Network (RNN) model, using any feature engineering. For improving the performance of the system, they further came up with a new hybrid model that is built of combining the RNN model and a similarity-based retrieval model. This model consists of a set of encoders and decoders. It also includes a classifier, to determine the significance of a numerical quantity and proposed a TF-TDF similarity-based retrieval model, to predict the question associated. To examine the performance of both models an experiment was conducted on a large set of data and surprisingly these outperform all the state-of-the art models built on statistical learning methods.

RNN models are generally used to perform Seq2Seq modelling. Though these models provide satisfying results over both small and large datasets, traditional ML models perform better on smaller datasets, due to the high lexical similarities [39].

3. Proposed Method

3.1. Problem Formulation

A single operation, single equation and single step arithmetic word problem P can be defined as a sequence of n words {w0, w1,...,wn-1} which contains a set of quantities QP = {q0, q1,….,qx-1}, where, n>x. The quantities i.e., the numeric values, appear in the quantity set according to the order of appearance of the numerical entities in P [10]. The set of relevant quantities can be defined as $Q_{P(rel)} = \{q_s, q_t\}$, where, $\{q_s, q_t\} \in Q_P$, i.e. $Q_{P(rel)} \subseteq Q_P$.

Let, $P_{SingleOp}$ is a set of arithmetic word problems, and each problem $P \in P_{SingleOp}$ can be solved by evaluating a correct mathematical equation E, which was formulated by the quantities of $Q_{P(rel)}$ and by selecting one of the operators op ∈ {+, -, *, /}. The equation E, for the problem $P \in P_{SingleOp}$ can be formulated by applying one of the possible equation format $\{E_{addmul}, E_{subdiv}\}$, described in section 3.4.

3.2. System Overview

Fig 1 describes the overview of the system. The detailed workflow is explained in the following sections.

3.3. Operation Prediction

Predicting the operation of the MWP is one of the major tasks. We used a multilevel classification framework like [10] for this task. In the level-1 classification framework, we manually studied the characteristics of each problem, and according to the characteristics we tried to broadly categorize them. Four such categories as we mentioned before are- Change, Combine, Compare and Division-Multiplication. The first three categories basically belong to Addition-Subtraction type problems and the last one refers to the entire Division-Multiplication type problems. These categories are further categorized into sub-categories in level-2 classification framework to determine the desired operation. We basically used keyword-based cues, positional cues, phrase cues, and pattern cues, etc., to classify the word problems in multiple levels. We are indebted to [10] as we have reused some of their identified features and reused them in our proposed method with various new cues and features.

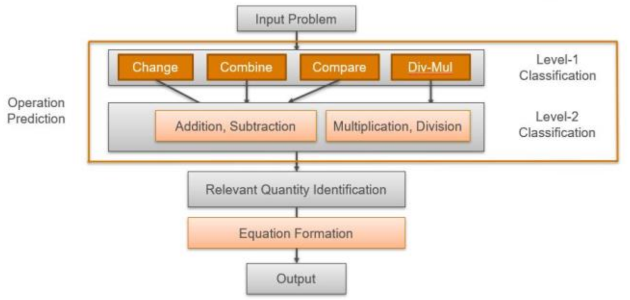

Fig 1. System overview of rule-based math word problem solver

3.3.1. Level-1 Classification Framework

I. Change

The category 'Change' can be defined as set of actions that causes the increment or decrement to the quantity belonging to a particular entity or variable.

- **Change Verb Keywords**- "gives", "takes", "loses", "lost", "add", "join", "left", "shares", "eaten" etc.
- **Change Non-Verb Keywords**- "now", "change", "sum", "away", "rest", "off", "empty" etc.

II. Combine

The category 'Combine' refers to the word problems which are related to the combination or collection of two or more entities. In this type of word problems, either the combined numerical value of participating entities is asked or the combined value and one of the participating entity's values are given and the other participating entity's value is being asked.

- **Combine Keywords**- "all", "total", "together", "altogether" etc.

III. Compare

The category 'Compare' represents the set of questions, where one quantity is being compared to another quantity. Here, the category does not always mean the comparison between two different entities, it could be the comparison to current numerical values associated to the state of same entity also. For example, "*Brenda starts with 7 Skittles. She buys 8 more. How many Skittles does Brenda end with?*" where, the additional quantity 8, is compared to the current numerical state of the entity, which is 7, to find out the actual answer.

- **Comparative Adjectives or Adverbs**- Any word belonging to the mentioned Part-of- speech tags (POS) like, "more", "less", "longer", "heavier", "fewer" etc.
- **Associated Comparative Keywords**- "another", "than" etc.

IV. Division-Multiplication

This category contains all the word problems of type 'Division and Multiplication'. Therefore, the features to identify this category are quite different than the previous three. Multiplication is about combining equal parts to make a whole and Division is about separating into equal parts. 'Division-Multiplication' are the key operations for some specific type of word problems as well, such as calculating time and distance, calculating area, etc.

- **"Equal part" Related Keywords-** "each", "every", "per".
- **Time & Distance Related Keywords-** "mile", "kilometre", "meter", "minute", "hour" etc.
- **Miscellaneous Div-Mul Keywords-** "whole", "times", "row", "split", "divide", "cost", "square", "cover", "do", "feet," etc.
- **Combined Div-Mul Keywords-** <"sold" and "does">, <"shares" and "among">.

By analysing the dataset, we observed that, one word problem may belong to multiple categories simultaneously, i.e., it may have keywords representing two different categories. For example, "*There are 4 marbles. 7 marbles **more** are added. How many are there **total**?*" Here, the keywords "more" and "total" appear together in a single word problem, where, "more" keyword is generally used to identify "Compare" type problems and "total" keyword indicates "Combine" type problems. To avoid this kind of conflicts, we prioritized the categories based on the number of occurrences of the keywords belonging to each category. Since the keyword "more" occurs more frequently to identify the category "Compare" with respect to "total" for "Combine", the priority is given to "Compare". Therefore, the above-mentioned problem is categorized as "Compare" type. Basically, we adopted the **precedence rules for different categories from the work of** [10]. According to these rules, "Compare" has the highest priority, followed by "Division-Multiplication", "Combine" and "Change" respectively.

- **Dealing with Overlapping Keywords-** As we know that, "Compare", "Combine" and "Change" are the categories of Addition-Subtraction type problems, their overlapping keyword features can easily be solved by applying the precedence of the categories. However, some exceptions may happen, in case of Division-Multiplication. Since it is possible to overlap some keywords between Addition-Multiplication and Subtraction-Division, there is a possibility of categorizing "Combine" or "Change" category problems as "Division-Multiplication" type, as "Division-Multiplication" has higher precedence than "Combine" and "Change". We have handled these cases explicitly. For example, "Linda has 34 candies. Chloe has 28. How many candies do they have in all?" Here, the keyword "do" belongs to "Division-Multiplication" category, whereas the keyword "all" belongs to "Combine" category. To overcome such conflicts, if we simply follow the precedence table, it returns its category as "Division-Multiplication" type, which is not correct. Thus, to identify the categories uniquely, we must follow some combined keyword features. The below mentioned list displays all such conditions.
- **Combined Explicit Combine Features-** If, the keywords (i) < "do" and "all"> (ii) < "do" and "altogether" > both present in the word problem, the problem should belong to the category "Combine".
- **Combined Explicit Change Features-** If, the keywords (i) < "each" and "added" > (ii) < "miles" and "left" > (iii) < "costs" and "change" > are both present in the word problem, it should be a part of "Change" type problem.
  The keywords, both of which appeared to be single, as well as combined features, the precedence of execution of these features should be combined feature followed by single feature. For example, the combined feature, < "costs" and "change"> has higher precedence than the single feature "costs".

3.3.2. Level-2 Classification Framework

For, level-2 classification, we reused the features identified in level-1 classification along with some extra features, which includes, keyword-based cues, positional cues, phrase cues, pattern cues, and the combination of these cues. To apply the positional cues, we have divided the input problem into two parts- "story part" that contains all the sentences excluding the question sentence, and "query part" that contains only the question sentence. The main objective of level-2 classification is to apply the unique features or combination of features on level-1 classification output to identify whether the input question performs any one operation among Addition, Subtraction, Multiplication, and Division.

Addition-Subtraction

I. Compare

According to the concept discussed in Level-1 Classification Framework (III. Compare) and by analysing the "Compare" category [31, 8] and the "Comparison" category [9], we have divided it into two subcategories- Comparative Addition and Comparative Subtraction.

- **Keyword Based Cues**
  (i) Presence of keyword "some" in the "Compare" type question always indicates "Subtraction" operation. (ii) Presence of keyword "another" always indicates "Addition" operation, according to the dataset.
- **Keyword Positional Cues-** If, the Comparative Adjectives or Adverbs are present in the "query part", the operation should be "Subtraction".
- **Combined Cues-** If, the Comparative Adjectives or Adverbs are present in the "story part", whether the operation is "Addition" or "Subtraction" is decided based on some other cues, **(i)** If, the comparison is done between two different entities and therefore, the keyword "than" is present in the question: At this situation, to identify, which entity is being compared with respect to another entity, keyword cues or positional cues are not sufficient. For example, if we consider the questions, (a) "*Ethan has 31 presents. Alissa has 22 more than Ethan. How many presents does Alissa have?*" and (b) "*Sean has 223 whistles. He has 95 more whistles than Charles. How many whistles does Charles have?*" both the questions seem similar according to the keyword and positional cues, but clearly they perform different operations. To overcome this scenario, we need to use **pattern cues** here. **(ii)** If the comparison is done to the numerical value associated with the current state of the same entity, the operation should be "Addition". Algorithm 1 shows the procedure.

Algorithm 1: compare_type_pattern_cue (question, predicted_category)

Input: (i) Word problem after lower casing the text (ii) Category of the input problem, which is the output of Level 1 classification.
Output: The predicted operation of the word problem.

```
predicted_operation ← ϕ
persons[] ← ϕ
if(predicted_category == "compare" and "than" ∊ question), then,
    resolve the co-references of the question.
    if(proper noun ∊ question),
        if(proper noun ∉ persons[]),
            persons[] ← P, where, P={P0, P1} is proper noun
    if("than" → next == P0), then,
        return(predicted_operation ← "addition")
    else,
        return(predicted_operation ← "subtraction")
```

II. Combine

As per the discussion of Level-1 Classification Framework (II. Combine), we divided it into two sub-categories, Combine Addition and Combine Subtraction.

- **Keyword Based Cues-** Presence of keyword "total" in the "Combine" type problems, always indicates "Addition" operation.
- **Keyword Positional Cues**
  (i) If, the keyword "all" is present in the "story part", along with the keyword "will" in the question, it indicates "Addition" operation, but, in that case, absence of the "will" keyword indicates "Subtraction" operation. (ii) If, the keyword "together" is present in the question, the operation should be "Addition". (iii) Presence of keyword "all" or "altogether" in the "query part" of "Combine" type question, always indicates "Addition" operation. (iv) If, the keyword "altogether" is present in the "story part", the operation should be "Subtraction".

III. Change

According to the concept discussed in Level-1 Classification Framework (I. Change), irrespective of the position of unknown quantity, we have tried to find out the features, that ultimately responsible for defining its sub-categories as, Change Addition and Change Subtraction.

- **Keyword Based Cues**
  (i) Presence of the keyword "sum" in the question, always indicates "Addition" operation. (ii) Presence of any one of the keywords "away", "empty", "rest", "loses", "lost", "change", "take", "off", "shares", "eaten", "gives" in the "Change" type question, always indicates "Subtraction" operation.
- **Keyword Positional Cues-** (i) If the keywords "added" or "join", is present in the "story part", the operation should be "Addition", otherwise, if these keywords are present in the "query part", the operation should be "Subtraction". (ii) If the keyword "left" is present in the "story part" and its part-of-speech is verb, the operation should be "Addition", else, if the keyword is present in the "query part", the operation should be "Subtraction".
  Division-Multiplication-As per the discussion of Level-1 Classification Framework (IV. Division-Multiplication), we have divided it into two subcategories- Division and Multiplication. We have already observed that, in many cases "Division" and "Multiplication" share same set of keywords, however based on several other factors, the ultimate operation is determined.
- **Combined Division-Multiplication Cues-(i)** If the keyword "each" is present in the "query part", and if, the "story part" of the question is null, the operation should be "Multiplication". Otherwise, the operation should be "Division". (ii) If the keyword "per" is present in the question, and
  -If the keywords "far", "miles", "points" etc. are present in the "query part" i.e., if the question is mainly asking about distance related information, the operation should be "Multiplication".
  -If the keywords "long", "minutes", "gallons" etc. are present in the "query part" i.e., if the question is mainly asking about time related information, the operation should be "Division". (iii) If the keyword "cost" is present in the question, and
  -If the phrase "how much" is present in the "query part", the operation should be "Multiplication".
  -If the phrase "how many" is present in the "query part", the operation should be "Division". (iv) If the keyword "times" is present in the "story part", the operation should be "Multiplication".
  (v) If the keyword "times" is present in the "query part", and
  - If, the keyword "will" is present in the question, the operation should be "Multiplication".
  - Otherwise, the operation should be Division. **(vi)** If the keyword "each" or "every" is present in the "story part", and
  -If, a numeric value is present in the "query part", the operation should be "Multiplication".
  -If the "Combine keywords" i.e., "all", "total", "altogether" are present in the question, the operation should be "Multiplication".
  - If none of the above two conditions is satisfied, there exists some multiplication and division problems, which are indistinguishable based on the keyword cues. For example, Table 2 lists up a few such cases and Algorithm 2 describes the rules we propose to handle them.

Table 2. Comparing item name in "story part" and "query part" to identify the final operation

Word Problem	Item Name in "Story Part"	Item Name in "Query Part"	Operator
Case 1: **(a) Marlee has 12 guests coming to her Halloween party. Each table will hold 3 guests. How many tables will she need?**	(a) table	(a) table	(a) "/"
(b) Michelle has 7 boxes of crayons. Each box holds 5 crayons. How many crayons does Michelle have?	(b) box	(b) crayon	(b) "*"
Case 2: **(c) Mrs. Heine is buying Valentine's Day treats for her 2 dogs. If she wants to buy them 3 heart biscuits each, how many biscuits does she need to buy?**	(c) biscuit	(c) biscuit	(c) "*"
(d) There are 14240 books in a library. They are arranged on shelves that hold 8 books each. How many shelves are in the library?	(d) book	(d) shelve	(d) "/"

Here, the underlying pattern of the questions play an important role to identify the final operation. The Algorithm 2 describes the same.

Algorithm 2: divmul_type_pattern_cue (story_part, query_part, predicted_category)

Inputs: (i) Story part of an input question. (ii) Query part of an input question. (iii) Category of the input problem, which is the output of Level 1 classification.
Output: The predicted operation of the word problem.

```
1. predicted_operation ← ϕ
2. item_name ← ϕ
3. if(predicted_category == “div-mul” and <“each”/ “every”> ∊ story_part), then,
4.     find the index of “each” / “every”
5.     if( noun(s) present between each_index+1 and end of story_part),
6.         item_name ← item_name + noun
7.     find out noun phrases ∊ query_part
8.     find out noun phrase that contains wh-word.
9.     find out the rightmost noun present in the wh-noun phrase.
10.    if(item_name ≠ ϕ), then,
11.        find out the rightmost noun present in the item_name
12.        if(item_name_rightmost_noun ==
           wh_phrase_rightmost_noun),then,
13.            return(predicted_operation ← “division”)
14.        else,
15.            return(predicted_operation ← “multiplication”)
16.    else,
17.        if(each_index-1 == noun), then,
18.            item_name ← lemmatized(each_index → prev)
19.            if(item_name_rightmost_noun ==
                wh_phrase_rightmost_noun),then,
20.                return(predicted_operation ← “multiplication”)
21.            else,
22.                return(predicted_operation ← “division”)
```

- **Explicit Division-Multiplication Keyword Cues-** (i) Presence of the keywords “do”, “cover”, “far”, “row”, “will” etc. in the question mostly indicate “Multiplication” operation. (ii) If, the keyword “whole” is present in the question, and the keyword “cover” is not present in the question, the operation should be “Division”. (iii) Presence of the keywords “split” “sold”, “fast” etc. in the Division-Multiplication type question, indicate the operation “Division”.

3.4. Identifying Relevant Quantities

After predicting operation for a word problem, the next challenging work is to identify the relevant quantities which are responsible for final answer generation. Basically, a word problem may contain irrelevant quantities. However, identifying irrelevant sentences seems simpler than identifying irrelevant quantities. Here, the irrelevant information does not only mean out of context information, but also the information, that is important for the problem definition, but not taking part in answer generation.

By analysing the dataset, we observed that, irrelevant information (or quantities) belongs to the word problems comprising of all the four types of operations i.e., “Addition”, “Subtraction”, “Multiplication” and “Division”. Since the features to identify all these operations are different, they are handled in different manner in identifying irrelevant quantities. Depending on the characteristic of the questions containing irrelevant information, we have divided these into three groups- “Addition-Subtraction”, “Division” and “Multiplication” and propose specific independent rules to filter out the irrelevant information (or quantities).

Pre-processing is considered as the most important step for the information extraction from the arithmetic word problems in the first step of pre-processing. We perform co-reference resolution and substitution to substitute the pronouns with the relevant nouns and used NeuralCoref [40] for this purpose. After that, the input text is segregated into two parts, ‘story part’ and ‘query part’. Further, the story part is divided into individual sentences. Then, we eliminated the conjunctions which are responsible for joining two quantities and re-constructed the sentences. We used SpaCy’s dependency parser [41] for this purpose. For example, if a question contains the sentence, “*Carolyn starts with 47 marbles and 6 oranges.”,* it is re-phrased as, “*Carolyn starts with 47 marbles*” and “*Carolyn starts with 6 oranges.*”

After pre-processing, we extracted the information provided in the “query part”. We observed that, the information ranges to four parameters depending on the type of operation the question belongs to. The parameters include **location**, **primary entity**, **person(s) involved** and **secondary entity.** It is not necessary that all the problems should have all these parameters. For example, “*There are 8 apples in a pile on the desk. Each apple comes in a package of 11. 5 apples are added to the pile. How many apples are there in the pile?*”, if we consider the query part, “pile” is considered as location and “apples” is considered as primary entity. Here, location means the place, not any geographical location. So, it is identified by matching the POS pattern, determiner followed by a preposition and a noun, and then by extracting only the noun from the matched pattern phrase. Likewise, primary entity is the entity, about which the problem is asking about, and secondary entity is another entity apart from the primary entity and are identified by extracting nouns from noun phrases and person(s) name are identified by extracting proper nouns from the “query part”.

Addition-Subtraction

The information extracted from the “query part” of Addition-Subtraction type problems, mainly consists of three types of information i.e., location, person(s) involved and primary entity. Combination of these is also possible. (i) If, the location information and the primary entity information, both are present in the “query part”, then,

-First search for the location in the sentences of “story part”. If, a sentence contains the location, then search for the presence of primary entity. If, the primary entity is also there, then only the quantity belongs to that sentence is considered as relevant. However, if the primary entity is not present in the sentence, since, the location information is present, then also the quantity present in the sentence is considered relevant by mapping it to the entity name of the previously qualified sentence. Hence, location has the higher precedence than the entity name. (ii) If, the location information is not present in the “query part”, but primary entity and person name(s) related information are present, then,

-First search for the person’s name(s) in the sentences of “story part”. If, a sentence contains the person’s name, then search for the presence of primary entity. If the primary entity is also present in the sentence, then only the quantity belonging to that sentence is considered as relevant. However, if the primary entity is not present in the sentence and only person name is present, then the quantity present in the sentence is considered relevant by mapping it to the entity name of the previously qualified sentence, if, no other entity name belongs to the same sentence. Hence, person name(s) has the higher priority than the entity name. (iii) If only the person’s name(s) related information is

present in the "query part", then,
-Consider the quantities relevant, which belong to the sentences containing the person's name(s), same as the person's name(s) present in the "query part". (iv) If only the primary entity related information is present in the "query part", then,
-Consider the quantities relevant, which belong to the sentences containing the entity, same as the entity information present in the "query part".

Division

By analysing the problems of division, we have observed that, the information extracted from the "query part", consists of only two types of information, i.e., primary entity and secondary entity. A quantity belonging to a sentence of "story part" is considered relevant, if, any of the entities of "query part" is present in that sentence. Hence, both entities are of equal priority.

Multiplication

The concept of identifying relevant quantities for multiplication type operation is a bit complex than the previous two types, as we have observed that, multiplication type questions may contain more than two quantities, though the third quantity is not actually extraneous information. For example, "*Tammy drove 55 miles in one hour. At that rate, how far can she drive in 36 hours?*" Here, "*one hour*" is not extraneous information, but not also the quantity which is responsible for final output generation.
The 'query part' of Multiplication type problems with irrelevant information, generally consists of two types of information, i.e., primary entity and person(s) name, but only primary entity name is sufficient for relevant information identification. A quantity belonging to a sentence of "story part" is considered relevant, if the primary entity of "query part" is present in that sentence. Therefore, primary entity has higher priority than the person(s) name in multiplication.

3.5. Equation Formation

It is the last step to generate final output. During forming the equation, determining the order of quantities is an important factor for "Division" and "Subtraction" type problems. So many times, numerical reasoning is required for this purpose. Since our dataset contains the word problems suitable for 2nd or 3rd grade students, no complex logic is needed for determining their order. Division type equation is formed by considering the larger number as dividend and the smaller number as divisor. Similarly, for "Subtraction" type question also, the smaller number is subtracted from the larger number. However, for "Addition" and "Multiplication" type problems, there is no issues for determining the order of quantities.
$E_{addmul} = Q_{P(rel)1}$ (op) $Q_{P(rel)2}$, where, op ∈ {+, *} and $Q_{P(rel)1}$, $Q_{P(rel)2}$ are the order of appearance of relevant quantities in $Q_{P(rel)}$
$E_{subdiv} = Q_{P(rel)L}$ (op) $Q_{P(rel)S}$, where, op ∈ {-, /} and $Q_{P(rel)L}$, $Q_{P(rel)S}$ are the larger and smaller quantities respectively from the set of relevant quantities $Q_{P(rel)}$.

4. Dataset and Performance Evaluation

In our proposed method, we used the Illinois SingleOp dataset, published by [2]. Most of the problems of this dataset were collected from [34]. The dataset contains the word problems, that covers all the basic four type of operations i.e., addition, subtraction, multiplication, and division. These word problems are basically designed by keeping in mind the analytical abilities of the 1st to 3rd grade students. Thus, the complexity of the word problems is limited up to single equation and single operation. However, they included the word problems with irrelevant information also to increase the complexity a step ahead. The dataset contains 562-word problems, consisting of 159 addition, 159 subtraction, 117 multiplication and 127 division type problems. Table 3 and Table 4 shows the performance of the proposed method in predicting the desired operation.

Table 3. Performance of Operation Prediction for Each Operation.

Operation	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
Addition	97.5	98.65	92.45	95.44
Subtraction	98.04	99.33	93.71	96.44
Multiplication	97.68	94.82	94.01	94.41
Division	99.28	99.2	97.64	98.41

4.1. Critical Discussion

From Table 5, we can observe that, our method outperforms most of the state-of-the-art systems built on the same dataset. The main idea behind the systems like KAZB [15], Roy and Roth [2] is to use classification techniques for different components of their systems, while the system proposed by [35], is a meaning-based system.

Table 4. Result of Our Method Specific to Each Operation.

Operation	Total Problems	Answers Correctly	Accuracy (%)
Addition	159	142	90.56
Subtraction	159	145	92.45
Multiplication	117	107	91.45
Division	127	124	97.63
Total	562	522	93.02

Table 5. Accuracy of other state-of-the-art Systems on SingleOp Dataset

Systems	SingleOp Dataset Accuracy (%)
KAZB [15]	73.7
Roy and Roth [2]	73.9
Tag-based [35]	79.5
AMWPS [10]	94.48
Our System	93.02

Our method uses rule-based approaches for both predicting operations and identifying relevant quantities. Although rule-based systems are expensive and difficult to construct for wide coverage of the rule set, their performance is always better than that of systems using purely statistical techniques. The possible reasons for the good performance of our method are listed below:

- Performing the categorization technique reduces the effort required to identify the operation associated with a problem.
- To identify the relevant operations, we thoroughly analyse the dataset to find out important keywords whose presence helps to uniquely identify the operations.
- In addition to the keywords, we have also tried to identify various patterns which in turn help to determine the operation.
- We use effective natural language processing techniques

like POS tagging, dependency parsing, shallow semantic parsing, co-reference resolution etc., to simplify the structure of the question and establish the relationship between the entities. These techniques play an important role in identifying relevant variables.

However, the performance of the current method is slightly lower than that proposed by [10], because the method of [10] is a hybrid method (a combination of rule-based and machine learning), in which several new concepts have been introduced, such as an object-oriented approach in modelling word problems belonging to different categories, RDBMS-based information storage, etc. Although it is ahead of our method in performance, its structure is quite complex, while our method works well with a simple system structure.

4.2. Error Analysis

The proposed method has produced 40 errors on SingleOp dataset. Out of these, 32 are due to either being unable to predict any operation or predicting wrong operation, and the rest 8 are due to the problem of identifying relevant quantities. Since, the method is rule-based, if a problem does not fit under any rule, it is not able to predict any operation. The origins of errors are discussed below.

- **Lack of world knowledge-** 9 such cases are there, where, to predict the operation of a problem or to identify the relevant quantities, real world knowledge is required. For example, "*There were 105 parents in the program and 698 pupils, too. How many people were present in the program?*" To solve this, the method must have the knowledge that "parents" and "pupils" are "people".
- **Lack of keyword cues-** 10 such cases are there, where no definite cues are present to identify the ultimate operation of the problem. For example, "*Misha has 34 dollars. How many dollars does she have to earn to have 47 dollars to buy a dog?*"
- **Lack of numerical reasoning-** Three such cases are there, where, to identify the relevant quantities present in the problem, only the keyword or pattern cues are not sufficient, some sort of numerical reasoning is also required. For example, "*Theresa has 32 crayons. Janice has 12 crayons. She shares 13 with Nancy. How many crayons will Theresa have?*" The co-reference resolver, NeuralCoref identifies "Janice" as the antecedent of "she", but the actual antecedent should be "Theresa".
- **Overlapped rule-based cues-** Nine such cases are there, where the method fails to predict right operation of a problem due to falling under incorrect rule.
- **Logical errors-** Two such cases are there, where the method fails to identify relevant quantities, due to logical error. Four more errors occur due to the word problems that were characteristically different than the word problems, inappropriate question structure, etc.
- **Wrongly identified POS tags-** Three such cases are there, where the method fails, due to wrong identification of Part-of-Speech by the spaCy's POS tagger [42]. For example, "*Emily collects 63 cards. Emily's father gives Emily 7 more. Bruce has 13 apples. How many cards does Emily have?*" Here, the POS tagger, returns the POS of "Emily" (query sentence) as "ADV" or adverb, therefore, unable to identify the person's name.

5. Conclusion and Future Work

In this paper, we present several algorithms built on multiple rules to solve the word problems with one equation and one operation belonging to the SingleOp dataset. Our work focuses on identifying important features and establishing relationships and dependencies among them to solve the word problems step by step. However, the main challenge was to identify the relevant quantities that are important for the final generation of the answer.

The proposed method performs relatively well in both predicting the operations and identifying the relevant quantities, although performance deteriorates in the case of incorrect identification of the relevant quantities due to errors in predicting the operations and the lack of other inferences. Operation prediction works more accurately for division-type problems (see Table 4) because most division-type input problems contain fewer ambiguous clues and the problems do not require additional background knowledge to be solved. However, notwithstanding any flaws inherent in the method, it outperforms most work published on the same dataset. Although the performance of our method is quite impressive, it is completely dependent on the features and rules created by hand. Our work can be further extended in numerous ways, as explained below.

- The hand-generated features could be used to train a classifier that automatically predicts the function of a word problem using a machine learning approach.
- A numerical inference module could be introduced to improve the algorithm's ability to identify relevant quantities, thus avoiding the errors caused by incorrect resolution of co-references.
- An inference module for world knowledge could also be extensively integrated into our method. The main purpose of this module is to deal with problems that require additional background knowledge to solve the problem.
- The concept of intelligent explanation of the solution could also be implemented and this module will show the solutions step by step.

References

[1] L. Verschaffel, B. Greer, and E. De Corte. *Making sense of word problems*. Leiden, Netherlands: Lisse Swets and Zeitlinger, 2000, doi:10.1023/A:1004190927303.

[2] S. Roy and D. Roth. Solving general arithmetic word problems. in Proc. 2015 Conf. Empirical Methods Natural Language Processing (EMNLP), Lisbon, Portugal, Sep. 17–21, 2015, pp. 1743-1752, doi:10.18653/v1/D15-1202.

[3] M. J. Nathan. Knowledge and situational feedback in a learning environment for algebra story problem solving. *Interactive Learn. Environ.* vol. 5, no. 1, pp. 135–159, 1998, doi:10.1080/1049482980050110.

[4] D. Arnau, M. Arevalillo-Herr´aez, L. Puig, and J. A. Gonz´alez-Calero. Fundamentals design and the operation of an intelligent tutoring system for the learning of the arithmetical and algebraic way of solving word problems. *Comput. & Educ.* vol. 63, pp. 119–130, Apr. 2013, doi:10.1016/j.compedu.2012.11.020

[5] D. Arnau, M. Arevalillo-Herr´aez, and J. A. Gonz´alez-Calero. Emulating human supervision in an intelligent tutoring system for arithmetical problem solving. *IEEE Trans. Learn. Technol.* vol. 7, no. 2, pp. 155–164, Apr./Jun. 2014, doi:

10.1109/TLT.2014.2307306.

[6] C. R. Beal. Animalwatch: An intelligent tutoring system for algebra readiness. in *Int. Handbook Metacognition Learn. Technologies.* Springer, Mar. 2013, pp. 337–348, doi:10.1007/978-1-4419-5546-3 22.

[7] M. S. Riley, J. G. Greeno, and J. I. Heller. Development of children's problem-solving ability in arithmetic. Univ. of Pittsburgh, Pittsburgh, PA, USA, Tech. Rep. LRDC-1984/37, 1984. [Online]. Available:https://files.eric.ed.gov/fulltext/ED252410.pdf

[8] C. R. Fletcher. Understanding and solving arithmetic word problems: A computer simulation. *Behav. Res. Methods, Instrum., & Comput*. vol. 17, no. 5, pp. 565–571, Sep. 1985, doi:10.3758/BF03207654.

[9] A. Mitra and C. Baral. Learning to use formulas to solve simple arithmetic problems. in Proc. 54th Annu. Meeting Association Computational Linguistics (ACL), Berlin, Germany, Aug. 7–12, 2016, pp. 2144–2153, doi: 10.18653/v1/P16-1202.

[10] S. Mandal and S. K. Naskar. Classifying and Solving Arithmetic Math Word Problems—An Intelligent Math Solver. in *IEEE Transactions on Learning Technologies*. vol. 14, no. 1, pp. 28-41, Feb. 2021, doi: 10.1109/TLT.2021.3057805.

[11] T. P. Carpenter, J. Hiebert, and J. M. Moser. Problem structure and first-grade children's initial solution processes for simple addition and subtraction problems. *J. Res. Math. Educ*., pp. 27–39, Jan. 1981, doi:10.5951/jresematheduc.24.5.0428.

[12] P. Nesher, J. G. Greeno, and M. S. Riley. The development of semantic categories for addition and subtraction. *Educational Stud. Math*. vol. 13, no. 4, pp. 373–394, Nov. 1982, doi:10.1007/BF00366618.

[13] G. Vergnaud. A classification of cognitive tasks and operations of thought involved in addition and subtraction problems. Addition subtraction: A Cogn. perspective, pp. 39–59, 1982, doi: 10.4324/ 9781003046585-4.

[14] T. P. Carpenter, E. Ansell, M. L. Franke, E. Fennema, and L. Weisbeck. Models of problem solving: A study of kindergarten children's problem-solving processes. *J. Res. Math. Educ*., pp. 428–441, Nov. 1993, doi:10.5951/jresematheduc.24.5.0428.

[15] N. Kushman, L. Zettlemoyer, R. Barzilay, and Y. Artzi. Learning to automatically solve algebra word problems. in *Proc. 52nd Annu. Meeting Association Computational Linguistics (ACL)*, Baltimore, MD, USA, Jun. 22–27, 2014, pp. 271–281, doi: 10.3115/v1/P14-1026.

[16] R. Koncel-Kedziorski, H. Hajishirzi.+ 90A. Sabharwal, O. Etzioni, and S. D. Ang. Parsing algebraic word problems into equations. *Trans. 01Assoc. Comput. Linguistics*. vol. 3, pp. 585–597, Dec. 2015, doi: 10.1162/tacl_a_00160.

[17] D.G. Bobrow. Natural language input for a computer problem solving system. 1964.

[18] E. Charniak. Computer Solution of Calculus Word Problem. 1968.

[19] Y. Bakman. Robust understanding of word problems with extraneous information. vol. arXiv preprint math/0701393, 2007.

[20] C. Liguda and T. Peffier. Modeling Math Word Problems with Augmented Semantic Networks. in *In: Bouma G., Ittoo A., Métais E., Wortmann H. (eds) Natural Language Processing and Information Systems. NLDB 2012.* vol. vol 7337, Springer, Berlin, Heidelberg., 2012, pp. 247-252, Lecture Notes in Computer Science.

[21] M.J. Hosseini, H. Hajishirzi, O. Etzioni, and N. Kushman. Learning to solve arithmetic word problems with verb categorization. in *In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014.*, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL., October 25-29,2014, pp. 523-533. [Online]. http://aclweb.org/anthology/D/D14/D14-1058.pdf

[22] S. Shi, Y. Wang, C. Lin, X. Liu, and Y. Rui. Automatically solving number word problems by semantic parsing and reasoning. in *In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, Lisbon, Portugal, September 17-21, 2015, pp. 1132-1142. [Online]. http://aclweb.org/anthology/D/D15/D15-1135.pdf

[23] S. Roy and D. Roth. Illinois math solver: Math reasoning on the web. in *In: Proceedings of the Demonstrations Session, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*, San Diego California, USA., June 12-17, 2016, pp. 52–56. [Online]. http://aclweb.org/anthology/N/N16/N16-3011.pdf

[24] S. Roy and D. Roth. Unit dependency graph and its application to arithmetic word problem solving. in *In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence.*, San Francisco, California, USA., February 4-9, 2017, pp. 3082–3088. [Online]. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14764

[25] S. Roy, T. Vieira, and D. Roth. Reasoning about quantities in natural language. vol. TACL 3, pp. 1–13, 2015. [Online]. https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/452

[26] S. Roy and D. Roth. Mapping to Declarative Knowledge for Word Problem Solving. *Transactions of the Association for Computational Linguistics*. vol. Volume 6, pp. 159-172, 2018.

[27] L. Zhou, S. Dai, and L. Chen. Learn to solve algebra word problems using quadratic programming. in *In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, Lisbon, Portugal, September 17-21, 2015, pp. 817-822.

[28] S. Upadhyay and M. Chang. Annotating derivations: A new evaluation strategy and dataset for algebra word problems. 2016. [Online]. http://arxiv.org/abs/1609.07197

[29] D. Huang, S. Shi, C. Lin, J. Yin, and W. 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, ACL 2016.* vol. Volume 1: Long Papers (2016), August 2016. [Online]. http://aclweb.org/anthology/P/P16/P16-1084.pdf

[30] Y. Wang, X. Liu, and S. Shi. Deep Neural Solver for Math Word Problems. pp. 845–854, January 2017. [Online]. https://www.aclweb.org/anthology/D17-1088.pdf

[31] M.S., et al. Riley. Development of children's problem-solving ability in arithmetic. 1984.

[32] D.J. Briars and J.H Larkin. An integrated model of skill in solving elementary word problems. vol. Cognition and instruction 1(3), pp. 245-296, 1984.

[33] D. Dellarosa. A computer simulation of childrens arithmetic word-problem solving. *Behavior Reaearch Methods*. vol. Instruments, & Computers 18(2), pp. 147-154, 1986.

[34] DadsWorksheets.com, Available at: https://www.dadsworksheets.com/worksheets/word-problems.html, accessed June 2021.

[35] C. Liang, S. Tsai, T. Chang, Y. Lin, and K. Su. A meaning-based English math word problem solver with understanding, reasoning and explanation. in *Proc. 26th Int. Conf. Computational Linguistics (COLING)*, Osaka, Japan, Dec. 11–16, 2016, pp. 151–155.

[36] Allen Institute for AI, Available at: http://allenai.org/data.html, accessed June 2021.

[37] S. Mandal and S. K. Naskar. Solving Arithmetic Mathematical Word Problems: A Review and Recent Advancements. ICITAM 2017: 95-114

[38] S. Mandal and S. K. Naskar. Solving Arithmetic Word Problems by Object Oriented Modeling and Query-Based Information Processing. Int. J. Artif. Intell. Tools 28(4): 1940002:1-1940002:23 (2019)

[39] S. Mandal, A. A. Sekh and S. K. Naskar. Solving arithmetic word problems: A deep learning based approach. J. Intell. Fuzzy Syst. 39(2): 2521-2531 (2020)

[40] NeuralCoref 4.0: Coreference Resolution in spaCy with Neural Networks, Available at: https://github.com/huggingface/neuralcoref, accessed June 2021.

[41] DependencyParser, Available at: https://spacy.io/api/dependencyparser, accessed June 2021.6

[42] Linguistic Features, Available at: https://spacy.io/usage/linguistic-features, accessed June 2021.

[43] Rule-based_Math_Word_Problem_Solver, Available at: https://github.com/Swagata-Acharya/Rule-based_Math_Word_Problem_Solver.git, accessed August 2021.