Computational Linguistics and Chinese Language Processing
Vol. 20, No. 2, December 2015, pp. 1-26

Designing a Tag-Based Statistical Math Word Problem Solver with Reasoning and Explanation

Yi-Chung Lin*, Chao-Chun Liang*, Kuang-Yi Hsu*,

Chien-Tsung Huang*, Shen-Yun Miao*, Wei-Yun Ma*,

Lun-Wei Ku*, Churn-Jung Liau*, and Keh-Yih Su*

Abstract

This paper proposes a tag-based statistical framework to solve math word problems with understanding and reasoning. It analyzes the body and question texts into their associated tag-based logic forms, and then performs inference on them. Comparing to those rule-based approaches, the proposed statistical approach alleviates rules coverage and ambiguity resolution problems, and our tag-based approach also provides the flexibility of handling various kinds of related questions with the same body logic form. On the other hand, comparing to those purely statistical approaches, the proposed approach is more robust to the irrelevant information and could more accurately provide the answer. The major contributions of our work are: (1) proposing a tag-based logic representation such that the system is less sensitive to the irrelevant information and could provide answer more precisely; (2) proposing a unified statistical framework for performing reasoning from the given text.

Keywords: Math Word Problem Solver, Machine Reading, Natural Language Understanding.

1. Introduction

Since *Big Data* mainly aims to explore the correlation between surface features but not their underlying causality relationship, the *Big Mechanism*[1] program was initiated by DARPA

* Institute of Information Science, Academia Sinica, 128 Academia Road, Section 2, Nankang, Taipei 11529, Taiwan
E-mail: {lyc; ccliang; ianhsu; joecth; jackymiu; ma; lwku; liaucj; kysu}@iis.sinica.edu.tw

[1] http://www.darpa.mil/Our_Work/I2O/Programs/Big_Mechanism.aspx

(from July 2014) to find out "*why*" behind the "Big Data". However, the pre-requisite for it is that the machine can read each document and learn its associated knowledge, which is the task of *Machine Reading* (MR) (Strassel *et al*., 2010). Therefore, the Natural Language and Knowledge Processing Group, under the Institute of Information Science of Academia Sinica, formally launched a 3-year MR project (from January 2015) to attack this problem.

As a domain-independent MR system is complicated and difficult to build, the *math word problem* (MWP) (Mukherjee & Garain, 2008) is chosen as the first task to study MR for the following reasons: (1) Since the answer for the MWP cannot be extracted by simply performing keyword matching (as Q&A usually does), MWP thus can act as a test-bed for understanding the text and then drawing the answer via inference. (2) MWP usually possesses less complicated syntax and requires less amount of domain knowledge. It can let the researcher focus on the task of understanding and reasoning, not on how to build a wide-coverage grammar and acquire domain knowledge. (3) The body part of MWP (which mentions the given information for solving the problem) usually consists of only a few sentences. Therefore, the understanding and reasoning procedure could be checked more efficiently. (4) The MWP solver could have its own standalone applications, such as computer tutor, etc. It is not just a toy test case.

According to the framework of making the decision while there are several candidates, previous MWP algebra solvers can be classified into: (1) Rule-based approaches with logic inference (Bobrow, 1964; Slagle, 1965; Charniak, 1968, 1969; Dellarosa, 1986; Bakman, 2007), which apply rules to get the answer (via identifying entities, quantities, operations, etc.) with a logic inference engine. (2) Rule-based approaches without logic inference (Gelb, 1971; Ballard & Biermann, 1979; Biermann & Ballard, 1980; Biermann *et al*., 1982; Fletcher, 1985; Hosseini *et al*., 2014), which apply rules to get the answer without a logic inference engine. (3) Purely statistics-based approaches (Kushman *et al*., 2014; Roy *et al.*, 2015), which use statistical models to identify entities, quantities, operations, and get the answer without conducting language analysis or inference.

The main problem of the rule-based approaches mentioned above is that the coverage rate problem is serious, as rules with wide coverage are difficult and expensive to construct. Also, it is awkward in resolving ambiguity problems. Besides, since they adopt Go/No-Go approach (unlike statistical approaches which can adopt a large Top-N to have high including rates), the error accumulation problem would be severe. On the other hand, the main problem of those approaches not adopting logic inference is that they usually need to implement a new handling procedure for each new type of problems (as the general logic inference mechanism is not adopted). Also, as there is no inference engine to generate the reasoning chain, additional effort would be required for generating the explanation. In contrast, the main problem of those purely statistical approaches is that they are sensitive to irrelevant

information (Hosseini *et al*., 2014) (as the problem is solved without first understanding the text). Also, the performance deteriorates significantly when they encounter complicated problems due to the same reason.

To avoid the problems mentioned above, a *tag-based statistical framework* which is able to perform understanding and reasoning is proposed in this paper. For each body statement (which specifies the given information), the text will be first analyzed into its corresponding semantic tree (with its anaphora/ellipses resolved and semantic roles labeled), and then converted into its associated logic form (via a few mapping rules). The obtained logic form is then mapped into its corresponding domain dependent generic concepts (also expressed in logic form). The same process also goes for the question text (which specifies the desired answer). Finally, the inference (based on the question logic form) is performed on the logic statements derived from the body text. Please note that a statistical model will be applied each time when we have choices.

Furthermore, to reply any kind of questions associated with the given information, we keep all related semantic roles (such as agent, patient, etc.) and associated *specifiers* (which restrict the given quantity, and is freely exchangeable with the term *tag*) in the logic form (such as verb(q1,進貨), agent(q1,文具店), head($n1_p$,筆), color($n1_p$,紅), etc.), which are regarded as various *tags* (or conditions) for selecting the appropriate information related to the given question. Therefore, the proposed approach can be regarded as a *tag-based statistical approach with logic inference*. Since extra-linguistic knowledge would be required for bridging the gap between the linguistic semantic form and the desired logic form, we will extract the desired background knowledge (ontology) from E-HowNet (Chen *et al*., 2005) for verb-entailment.

In comparison with those rule-based approaches, the proposed approach alleviates the ambiguity resolution problem (i.e., selecting the appropriate semantic tree, anaphora/co-reference, domain-dependent concepts, inference rules) via a statistical framework. Furthermore, our tag-based approach provides the flexibility of handling various kinds of possible questions with the same body logic form. On the other hand, in comparison with those purely statistical approaches, the proposed approach is more robust to the irrelevant information (Hosseini *et al*., 2014) and could provide the answer more precisely (as the semantic analysis and the tag-based logic inference are adopted). In addition, with the given reasoning chain, the explanation could be more easily generated. Last, since logic inference is a general problem solving mechanism, the proposed approach can solve various types of problems that the inference engine could handle (i.e., not only arithmetic or algebra as most approaches aim to).

The contributions of our work are: (1) Proposing a semantic composition form for abstracting the text meaning to perform semantic reasoning; (2) Proposing verb entailment via

E-HowNet for bridging the lexical gap (Moldovan & Rus, 2001); (3) Proposing a tag-based logic representation to adopt one body logic form for handling various possible questions; (4) Proposing a set of domain dependent (for math algebra) generic concepts for solving MWP; (5) Proposing a statistical solution type classifier to indicate the way for solving MWP; (6) Proposing a semantic matching method for performing unification; (7) Proposing a statistical framework for performing reasoning from the given text.

2. Design Principles

Since we will have various design options in implementing a math word problem solver, we need some guidelines to judge which option is better when there is a choice. Some principles are thus proposed as follows for this purpose:

(1) Solutions should be given via understanding and inference (versus the template matching approach proposed in (Kushman *et al*., 2014), as the math word problem is just the first case for our text understanding project and we should be able to explain how the answer is obtained.

(2) The expressiveness of the adopted body logical form should be powerful enough for handling various kinds of possible questions related to the body, which implies that logic form transformation should be information lossless. In other words, all the information carried by the semantic representation should be kept in the corresponding logical form. It also implies that the associated body logical form should be independent on the given question (as we don't know which question will be asked later).

(3) The dynamically constructed knowledge should not favor any specific kind of problem/question. This principle suggests that the *Inference Engine* (IE) should regard logic statements as a flat list, instead of adopting a pre-specified hierarchical structure (e.g., the container adopted in (Hosseini *et al*., 2014), which is tailored to some kinds of problems/questions). Any desired information will be located from the list via the same mechanism according to the specified conditions.

(4) The *Logic Form Converter* (LFC) should be compositional (Moldovan & Rus, 2001) after giving co-reference and solution type[2], which implies that each sub-tree (or nonterminal node) should be independently transformed regardless of other nodes not under it, and the logic form of a given nonterminal node is formed by concatenating the corresponding logic forms of its associated child-nodes.

(5) The IE should only deal with domain dependent generic concepts (instead of complicated

[2] Solution Type specifies the desired mathematic utility/operation that LFC should adopt (see Section 3.3 for details).

problem dependent concepts); otherwise, it would be too tedious. Take the problem "100 顆糖裝成 5 盒糖, 1 盒糖裝幾顆糖? (If 100 candies are packed into 5 boxes, how many candies are there in a box?)" as an example. Instead of using a problem-dependent *First Order Logic* (FOL) predicate like "裝成(100,顆,糖,5,盒,糖)", the problem-independent FOL functions/predicates like "quan(q1, 顆, 糖) = 100", "quan(q2, 盒, 糖) = 5", "qmap(m1,q1,q2)", and "verb(m1,裝成)" are adopted to represent the facts provided by problem description[3].

(6) The LFC should know the global skeleton of the whole given text (which is implicitly implied by the associated semantic segments linked via the given co-reference information) to achieve a reasonable balance between it and the IE.

(7) The IE should separate the knowledge from the reasoning procedures to ease porting, which denotes that those domain dependent concepts and inference rules should be kept in a declarative form (and could be imported from some separated files); and the inference rules should not be a part of the IE's source code.

3. System Framework

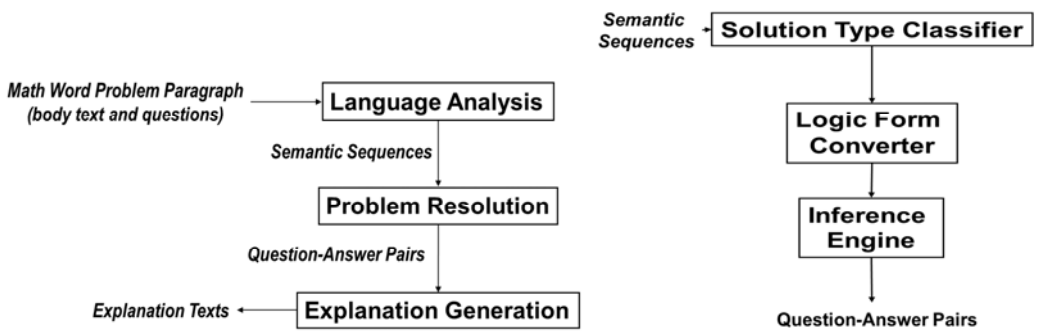

Figure 1.The block diagram of the proposed Math Word Problem Solver.

The block diagram of the proposed MWP solver is shown in Figure 1. First, every sentence in the MWP, including both body text and the question text, is analyzed by the *Language Analysis* module, which transforms each sentence into its corresponding *Semantic Representation* (SR) *tree*. The sequence of SR trees is then sent to the *Problem Resolution* module, which adopts logic inference approach to obtain the answer for each question. Finally,

[3] "quan(…)" is an FOL function to describe quantity facts. "quan(q1,顆,糖)=100" and "quan(q2,盒,糖)=5" describe two quantity facts about "100 顆糖" and "5 盒糖", respectively. "qmap(m1,q1,q2)" is an FOL predicate to describe that there is a relation (denoted by "m1") between two quantity facts "q1" and "q2". "verb(m1,裝成)" indicates that the verb "裝成" is associated with the quantity relation "m1".

the *Explanation Generation* module will explain how the answer is obtained (in natural language text) according to the given reasoning chain.

As the figure depicted, the Problem Resolution module in our system consists of three components: *Solution Type Classifier* (STC), LFC and IE. The STC suggests a scenario to solve the problem for every question in an MWP. In order to perform logic inference, the LFC first extracts the related facts from the given SR tree and then represents them as FOL predicates/functions (Russell & Norvig, 2009). It also transforms each question into an FOL-like utility function according to the assigned solution type. Finally, according to inference rules, the IE derives new facts from the old ones provided by the LFC. Besides, it is also responsible for providing utilities to perform math operations on related facts.

The entities (like noun phrases) or events (like verb phrases) described in the given sentence may be associated with modifiers, which usually restrict the scope (or specify the property) of the entities/events that they are associated. Since the system does not know which kind of questions will be asked when it reads the body sentences, modifiers should be also included in logic expressions (act as specifiers) and involved in binding. Therefore, the reification technique (Jurafsky & Martin, 2000) is employed to map the nonterminals in the given semantic tree, including verb phrases and noun phrases, into quantified objects which can be related to other objects via specified relations. For example, the logic form of the noun phrase "紅筆(red pens)" would be "color(n1,紅)&head(n1,筆)", where "n1" is an identified object referring to the noun phrase. Usually, the specifiers in the *Body Logic Form* (BLF) are optional in *Question Logic Form* (QLF), as the body might contain irrelevant text. On the contrary, the specifiers in the QLF are NOT optional (at least in principle) in BLF (i.e., the same (or corresponding) specifier must exist in BLF). This restriction is important as we want to make sure that each argument (which will act as a filtering-condition) in the QLF will be exactly matched to keep irrelevant facts away during the inference procedure.

Take the MWP "文具店進貨 2361 枝紅筆和 1587 枝藍筆(A stationer bought 2361 red pens and 1587 blue pens), 文具店共進貨幾枝筆(How many pens did the stationer buy)?" as an example. The STC will assign the operation type "Sum" to it. The LFC will extract the following facts from the first sentence:

quan(q1,枝,$n1_p$)=2361&verb(q1,進貨)&agent(q1,文具店)&head($n1_p$,筆)&color($n1_p$,紅)

quan(q2,枝,$n2_p$)=1587&verb(q2,進貨)&agent(q2,文具店)&head($n2_p$,筆)&color($n2_p$,藍)

The quantity-fact "2361 枝紅筆(2361 red pens)" is represented by "quan(q1,枝,$n1_p$)=2361", where the argument "$n1_p$"[4] denotes "紅筆(red pens)" due to the facts "head($n1_p$,筆)" and "color($n1_p$, 紅)". Also, those specifiers "verb(q1, 進 貨)&agent(q1, 文 具 店)&head($n1_p$, 筆)&color($n1_p$,紅)" are regarded as various tags which will act as different conditions for selecting the appropriate information related to the question specified later. Likewise, the quantity-fact "1587 枝藍筆(1587 blue pens)" is represented by "quan(q2,枝,$n2_p$)=1587". The LFC also issues the utility call "ASK Sum(quan(?q,枝,筆),verb(?q,進貨)&agent(?q,文具店))" (based on the assigned solution type) for the question. Finally, the IE will select out two quantity-facts "quan(q1, 枝 ,$n1_p$)=2361" and "quan(q2, 枝 ,$n2_p$)=1587", and then perform "Sum" operation on them to obtain "3948".

If the question in the above example is "文具店共進貨幾枝紅筆(How many red pens did the stationer buy)?", the LFC will generate the following facts and utility call for this new question:

head($n3_p$,筆)&color($n3_p$,紅)
ASK Sum(quan(?q,枝,$n3_p$),verb(?q,進貨)&agent(?q,文具店))

As the result, the IE will only select the quantity-fact "quan(q1,枝,$n1_p$)=2361", because the specifier in QLF (i.e., "color($n3_p$,紅)") cannot match the associated specifier "藍(blue)" (i.e., "color($n2_p$,藍)") of "quan(q2,枝,$n2_p$)=1587". After performing "Sum" operation on it, we thus obtain the answer "2361". Each module will be described in detail as follows (We will skip Explanation Generation due to space limitation. Please refer to (Huang *et al.*, 2015) for the details).

3.1 Language Analysis (Jurafsky & Martin, 2000)

Since the Chinese sentence is a string of characters with no delimiters to mark word boundaries, the first step for analyzing the MWP text is to segment each given sentence string into its corresponding word sequence. Our Chinese word segmentation system (Chen & Ma, 2002; Ma & Chen, 2003) adopts a modularized approach. Independent modules were designed to solve the problems of segmentation ambiguities and identifying unknown words. Segmentation ambiguities are resolved by a hybrid method of using heuristic and statistical rules. Regular-type unknown words are identified by associated regular expressions, and

[4] The subscript "p" in "$n1_p$" indicates that "$n1_p$" is a pseudo nonterminal derived from the nonterminal "n1", which has four terminals "2361", "枝", "紅" and "筆". More details about pseudo nonterminal will be given at Section 3.3.

irregular types of unknown words are detected first by their occurrence and then extracted by morphological rules with statistical and morphological constraints. Part-of-Speech tagging is also included in the segmentation system for both known and unknown words by using HMM models and morphological rules. Please refer to (Tseng & Chen, 2002; Tsai & Chen, 2004) for the details.

In order to design a high precision and broad coverage Chinese parser, we had constructed a Chinese grammar via generalizing and specializing the grammar extracted from Sinica Treebank (Hsieh *et al*., 2013; Hsieh *et al*., 2014) to achieve this goal. The designed F-PCFG (Feature-embedded Probabilistic Context-free Grammar) parser was based on the probabilities of the grammar rules. It evaluates the plausibility of each syntactic structure to resolve parsing ambiguities. We refine the probability estimation of a syntactic tree (for tree-structure disambiguation) by incorporating word-to-word association strengths. The word-to-word association strengths were self-learned from parsing the CKIP corpus (Hsieh *et al*., 2007). A semantic-role assignment capability is also incorporated into the system.

3.1.1 Semantic Composition

Once the syntactic structure (with semantic roles) for a sentence is obtained, its semantic representation can be further derived through a process of semantic composition (from lexical senses) and achieved near-canonical representations. To represent lexical senses, we had implemented a universal concept-representation mechanism, called E-HowNet (Chen *et al*., 2005; Huang *et al*., 2014). It is a frame-based entity-relation model where word senses are expressed by both primitives (or well-defined senses) and their semantic relations. We utilize E-HowNet to disambiguate word senses by referencing its ontology and the related synsets of the target words.

To solve math word problems, it is crucial to know who or what entity is being talked about in the descriptions of problems. This task is called reference resolution, and it can be classified into two types – anaphora resolution and co-reference resolution. Anaphora resolution is the task of finding the antecedent for a single pronoun while co-reference is the task of finding referring expressions (within the problem description) that refer to the same entity. We attack these two types of resolution mainly based on assessing whether a target pronoun/entity coincides its referent candidate in E-HowNet definition. For example, the definition of "她(she)" is "{3rdPerson|他人:gender={female|女}}". Therefore, it would restrict that the valid referent candidates must be a female human, and result in a much fewer number of candidates for further consideration.

In the following example, the semantic composition, anaphora resolution and co-reference resolution are shown in the table.

小豪 有 62 張 貼紙 ， 哥哥 再 給 他 56 張 ， 小豪 現在 共 有 幾張 貼紙 ？
(Xiaohao had 64 stickers, and his brother gave him 56 more. How many stickers does Xiahao have now?)

小豪有 62 張貼紙，	哥哥再給他 56 張，	小豪現在共有幾張貼紙？
{有(2): theme={[x1]小豪(1)}, range={貼紙(4): quantifier={６２張(3)} } } **小豪(1)**: {human|人:name={"小豪"}} **有(2)**: {own|有} **６２張(3)**: quantifier={張.null|無義:quantity={62}} **貼紙(4)**: {paper|紙張: qualification={sticky|黏}}	{給(3): agent={哥哥(1)}, time={再(2)}, goal={[x1]他(4)}, theme={貼紙(5.1): quantifier={５６張(5)} } } **哥哥(1)**: {哥哥|ElderBrother} **再(2)**: frequency={again|再} **給(3)**: {give|給} **他(4)**: {3rdPerson|他人} **５６張(5)**: quantifier={張.null|無義:quantity={56}} **貼紙(5.1)**: {paper|紙張:qualification={sticky|黏}}	{有(4): theme={[x1]小豪(1)}, time={現在(2)}, quantity={共(3)}, range={貼紙(6): quantifier={幾張(5)} } } **小豪(1)**: {human|人:name={"小豪"}} **現在(2)**: {present|現在} **共(3)**: {all|全} **有(4)**: {own|有} **幾張(5)**: quantifier={張.null|無義:幾.quantity={Ques|疑問}} **貼紙(6)**: {paper|紙張:qualification={sticky|黏}}

We use numbers following words to represent words'positions in a sentence. For instance, "有(2)" is the second word in the first sentence. The semantic representation uses a near-canonical representation form, where semantic role labels, such as "agent", "theme" and "range", are marked on each word, and each word is identified with its sense, such as "有(2): {own|有}".

The co-referents of all sentences in a math problem are marked with the same "x[#]". For example, we mark the proper noun "小豪(1)" with "[x1]" to co-refer with the pronoun "他(4)" and the second occurrence of the proper noun "小豪(1)". In the second sentence of the example, the head of the quantifier "５６張" is omitted in the text but it is recovered in the semantic representation and annotated with a decimal point in its word position. The missing head is recovered as "貼紙(5.1)", which is an extra word with its constructed position based on decimal point.

3.2 Solution Type Identification

However, even we know what the given math word problem means, we still might not know how to solve it if we have not been taught for solving the same type of problems in a math class before (i.e., without enough math training/background). Therefore, we need to collect various types of math operations (e.g., addition, subtraction, multiplication, division, sum, etc.), aggregative operations (e.g., Comparison, Set-Operation, etc.) and specific problem types (e.g., Algebra, G.C.D., L.C.M., etc.) that have been taught in the math class. And the LFC needs to know which math operation, aggregative operation or specific problem type should be adopted to solve the given problem. Therefore, we need to map the given semantic representation to a specific problem type. However, this mapping is frequently decided based on the global information across various input sentences (even across body text and question text). Without giving the corresponding mathematic utility/operation, the logic form transformation would be very complicated. A *Solution Type Classifier* (STC) is thus proposed to decide the desired utility/operation that LFC should adopt (i.e., to perform the mapping).

Currently, 16 different solution types are specified (in Table 1; most of them are self-explained with their names) to cover a wide variety of questions found in our elementary math word corpus. They are listed according to their frequencies found in 75 manually labeled questions. The STC is similar to the *Question Type Classifier* commonly adopted at Q&A (Loni, 2011). For mathematic operation type, it will judge which top-level math operation is expected (based on the equation used to get the final answer). For example, if the associated equation is "Answer = q1 – (q2 × q3)", then "Subtraction" will be the assigned math operation type, which matches human reasoning closely.

Table 1. Various solution types for solving elementary school math word problems with frequency in the training set (75 questions in total).

Multiply (24%)	Utility (6%)	Surplus (4%)	L.C.M (2%)
Sum (14%)	Algebra (5%)	Difference (4%)	G.C.D (2%)
Subtraction (12%)	Comparison (5%)	Ceil-Division (3%)	Addition (1%)
Floor-Division (7%)	Ratio (5%)	Common-Division (3%)	Set-Operation (1%)

Take the following math word problem as an example, "一艘輪船 20 分鐘可以行駛 25 公里(A boat sails 25 kilometers in 20 minutes)， 2.5 小時可以行駛多少公里(How far can it sail in 2.5 hours)？". Its associated equation is "Answer = 150 × (25÷20)". Therefore, the top-level operation is "Multiplication", and it will be the assigned solution type for this example. However, for the problem "某數乘以 11(Multiply a number with 11)， 再除以 4 的答案是 22(then divide it by 4. The answer is 22)， 某數是多少(What is the number)？", its

associated equation is "Answer×11÷4 = 22"; since there is no specific natural top-level operation, the "Algebra" solution type will be assigned[5].

The STC will check the SR trees from both the body and the question to make the decision. Therefore, it provides a kind of global decision, and the LFC will perform logic transformation based on it (i.e., the statistical model of the LFC is formulated to condition on the solution type). Currently, a SVM classifier with linear kernel functions (Chang & Lin, 2011) is used, and it adopted four different kinds of feature-sets: (1) all word unigrams in the text, (2) head word of each nonterminal (inspired by the analogous feature adopted in (Huang *et al.*, 2008) for question classification), (3) E-HowNet semantic features, and (4) pattern-matching indicators (currently, patterns/rules are manually created).

3.3 Logic Form Transformation

A two-stage approach is adopted to transform the SR tree of an input sentence to its corresponding logic forms. In the first stage, the syntactic/semantic relations between the words are deterministically transformed into their domain-independent logic forms. Afterwards, crucial generic math facts and the possible math operations are non-deterministically generated (as domain-dependent logic forms) in the second stage. Basically, logic forms are expressed with the first-order logic (FOL) formalism (Russell & Norvig, 2009)

In the first stage, FOL predicates are generated by traversing the input SR tree which mainly depicts the syntactic/semantic relations between its words (with associated word-senses). For example, the SR tree of the sentence "100 顆糖裝成 5 盒(If 100 candies are packed into 5 boxes)" is shown as follows:

{裝成(t1); theme={糖(t2); quantity=100(t3); unit=顆(t4)};
result={糖(t5); quantity=5(t6); unit=盒(t7)} }

Where "theme" and "result" are semantic roles, and information within brace are their associated attributes. Also, the symbols within parentheses are the identities of the terminals in the SR tree. Note that the terminal t5 is created via zero anaphora resolution in the language analysis phase. The above SR tree is transformed into the following FOL predicates separated by the logic AND operator &.

[5] However, the "Algebra" solution type in this case is useless to LFC because the body text has already mentioned how to solve it, and the LFC actually does not need STC to tell it how to solve the problem.

verb(v1,t1)&theme(v1,n1)&result(v1,n2)&
head(n1,t2)&quantity(n1,t3)&unit(n1,t4)&
head(n2,t5)&quantity(n2,t6)&unit(n2,t7)

All the first arguments of the above FOL predicates (i.e., v1, n1 and n2) are the identities to the nonterminals in the SR tree. To ease reading, the terminal identities in logic forms are replaced with their corresponding terminal strings in the rest of this paper. After replacement, the above logic forms become more readable as follows:

verb(v1,裝成)&theme(v1,n1)&result(v1,n2)&head(n1,糖)&quantity(n1,100)&
unit(n1,顆)&head(n2,糖)&quantity(n2,5)&unit(n2,盒)

The above FOL predicates are also called logic-form-1 (LF1) facts. The predicate names of LF1 facts are just the domain-independent syntactic/semantic roles of the constituents in a sub-tree. Therefore, the LF1 facts are also domain-independent.

The domain-dependent logic-form-2 (LF2) facts are generated in the second stage. The LF2 facts are derived from some crucial generic math facts associated with quantities and relations between quantities. The FOL function "quan(*quan_id, unit_id, object_id*) = number" is used to describe the facts about quantities. The first argument is a unique identity to represent this quantity-fact. The other arguments and the function value describe the meaning of this fact. For example, "qaun(q1,顆,糖) = 100" means "100 顆糖(100 candies)" and "qaun(q2,盒,糖) = 5" means "5 盒糖(five boxes of candies)". The FOL predicate "qmap(*map_id, quan_id*$_1$, *quan_id*$_2$)" (denotes the mapping from *quan_id*$_1$ to *quan_id*$_2$) is used to describe a relation between two quantity-facts, where the first argument is a unique identity to represent this relation. For example, "qmap(m1, q1, q2)" indicates that there is a relation between "100 顆糖" and "5 盒糖". Now, LF2 facts are transformed by rules with a predefined set of lexico-semantic patterns as conditions. When more cases are exploited, a nondeterministic approach would be required.

In additional to domain-dependent facts like "quan(…)" and "qmap(…)", some auxiliary domain-independent facts associated with *quan_id* and *map_id* are also created in this stage to help the IE find the solution. The auxiliary facts of the *quan_id* are created by 4 steps: First, locate the nonterminal (said n_q) which *quan_id* is coming from. Second, traverse upward from n_q to find the nearest nonterminal (said n_v) which directly connects to a verb terminal. Third, duplicate all LF1 facts whose first arguments are n_v, except the one whose second argument is n_q. Finally, replace the first arguments of the duplicated facts with *quan_id*. In the above

example, for the quantity-fact q1, n_q is n1 and n_v is v1 in the first and second steps. "verb(v1, 裝成)" and "result(v1, n2)" will be copied at the third step. Finally, "verb(q1, 裝成)" and "result(q1, n2)" are created. Likewise, "verb(q2, 裝成)" and "theme(q2, n1)" are created for q2. The auxiliary facts of "qmap(*map_id, quan_id*$_1$, *quan_id*$_2$)" are created by copying all facts of the forms "verb(*quan_id*$_1$, *)" and "verb(*quan_id*$_2$, *)" (where "*" is a wildcard), and then replace all the first arguments of the copied facts with *map_id*. So, "verb(m1, 裝成)" is created for m1.

Sometimes, the third argument of a quantity-fact (i.e., *object_id*) is a pseudo nonterminal identity created in the second stage. For example, the LF1 facts of the phrase "2361 枝紅筆 (2361 red pens)" are "quantity(n1, 2361)", "unit(n1, 枝)", "color(n1, 紅)" and "head(n1, 筆)", where n1 is the nonterminal identity of the phrase. A pseudo nonterminal identity, said $n1_p$, is created to carry the terminals "紅(red)" and "筆(pen)" so that the quantity-fact "2361 枝紅筆(2361 red pens)" can be expressed as "quan(q1, 枝, $n1_p$) = 2361". The subscript "p" in $n1_p$ indicates that $n1_p$ is a pseudo nonterminal derived from the n1. To express that fact that $n1_p$ carries the terminals "紅(red)" and "筆(pen)", two auxiliary facts "color($n1_p$, 紅)" and "head($n1_p$, 筆)" are also generated.

The questions in an MWP are transformed into FOL-like utility functions provided by the IE. One utility function is issued for each question to find the answer. For example, the question "文具店共進貨幾枝筆(How many pens did the stationer buy)" is converted into "ASK Sum(quan(?q,枝,筆), verb(?q,進貨)&agent(?q,文具店))". This conversion is completed by two steps. First, select an IE utility (e.g., "Sum(…)") to be called. Since the solution type of the question is "Sum", the IE utility "Sum(*function*, *condition*) = value" is selected. Second, instantiate the arguments of the selected IE utility. In this case, the first argument *function* is set to "quan(?q, 枝, 筆)" because an unknown quantity fact is detected in the phrase "幾枝筆 (how many pens)". Let the FOL variable "?q" play the role of *quan_id* in the steps of finding the auxiliary facts. The auxiliary facts "verb(?q, 進貨)" and "agent(?q, 文具店)" are obtained to compose the second argument *condition*.

To sum up, the LFC transforms the semantic representation obtained by language analysis into domain dependent FOL expressions on which inference can be performed. In contrast, most researches of semantic parsing (Jurcicek *et al*., 2009; Das *et al*., 2014; Berant *et al*., 2013; Allen, 2014) seek to directly map the input text into the corresponding logic form. Therefore, across sentences deep analysis of the input text (e.g., anaphora and co-reference resolution) cannot be handled. The proposed two-stage approach (i.e., language analysis and then logic form transformation) thus provides the freedom to enhance the system capability for handling complicated problems which require deep semantic analysis.

3.4 Logic Inference

3.4.1 Basic Operation

In our design, an IE is used to find the solution for an MWP. It is responsible for providing utilities to select desired facts and then obtaining the answer by taking math operations on those selected facts. In addition, it is also responsible for using inference rules to derive new facts from the facts directly provided from the description of the MWP. Facts and inference rules are represented in first-order logic (FOL) (Russell & Norvig, 2009).

In some simple cases, the desired answer can be calculated from the facts directly derived from the MWP. For those cases, the IE only needs to provide a utility function to calculate the answer. In the example of Figure 2, quantities 300, 600, 186 and 234 are mentioned in the MWP. The LFC transforms the question into "ASK Sum(quan(?q,朵,百合), verb(?q,賣出)&agent(?q,花店)" to ask the IE to find the answer, where "Sum(…)" is a utility function provided by the IE. The first argument of "Sum(…)" is an FOL function to indicate which facts should be selected. In this case, the unification procedure of the IE will successfully unify the first argument "quan(?q, 朵, 百合)" with three facts "quan(q2, 朵, 百合)", "quan(q3, 朵, 百合)" and "quan(q4, 朵, 百合)". When unifying "quan(?q, 朵, 百合)" with "quan(q2, 朵, 百合)", the FOL variable "?q" will be bound/substituted with q2. The second argument of "Sum(…)" (i.e., "verb(?q,賣出)&agent(?q,花店)") is the condition to be satisfied. Since "quan(q2, 朵, 百合)" is rejected by the given condition, "Sum(…)" will sum the values of the remaining facts (i.e., q3 and q4) to obtain the desired answer "420".

花店進貨 300 朵玫瑰和 600 朵百合(A flower store bought 300 roses and 600 lilies), 上午賣出 186 朵百合(It sold 186 lilies in the morning)，下午賣出 234 朵(It sold 234 lilies in the afternoon)，問花店共賣出幾朵百合(How many lilies did the flower store sell)?
quan(q1,朵,玫瑰)=300&verb(q1,進貨)&agent(q1,花店)&… quan(q2,朵,百合)=600&verb(q2,進貨)&agent(q2,花店)&… quan(q3,朵,百合)=186&verb(q3,賣出)&agent(q3,花店)&… quan(q4,朵,百合)=234&verb(q4,賣出)&agent(q4,花店)&… ASK Sum(quan(?q,朵,百合), verb(?q,賣出)&agent(?q,花店))

Figure 2. A simple problem and its essential corresponding logic forms.

Table 2 lists the utilities provided by the IE. The first one, as we have just described, returns the sum of the values of FOL function instances which can be unified with the function argument and satisfy the condition argument. The *Addition* utility simply returns the value of "$value_1$+$value_2$", where $value_i$ is either a constant number, or an FOL function value, or a value returned by a utility. Likewise, *Subtraction* and *Multiplication* utilities return

"$value_1$−$value_2$" and "$value_1$×$value_2$" respectively. *Difference* returns the absolute value of *Subtraction*. *CommonDiv* returns the value of "$value_1$÷$value_2$". *FloorDiv* returns the largest integer value not greater than "$value_1$÷$value_2$" and *CeilDiv* returns the smallest integer value not less than "$value_1$÷$value_2$". *Surplus* returns the remainder after division of $value_1$ by $value_2$.

Table 2. The utilities provided by the IE.

Sum(function, condition)=value	CommonDiv($value_1$, $value_2$)=value
Addition($value_1$, $value_2$)=value	FloorDiv($value_1$, $value_2$)=value
Subtraction($value_1$, $value_2$)=value	CeilDiv($value_1$, $value_2$)=value
Difference($value_1$, $value_2$)=value	Surplus($value_1$, $value_2$)=value
Multiplication($value_1$, $value_2$)=value	

Solving MWPs may require deriving new facts according to common sense or domain knowledge. In Figure 3, the MWP provides the facts that "爸爸(Papa)" bought something but it does not provide any facts associated to the money that "爸爸(Papa)" must pay. As a result, we are not able to obtain the answer from the question logic form "Sum(quan(?q,元,#), verb(?q,付)&agent(?q,爸爸))". However, it is common sense that people must pay some money to buy something. The following inference rule implements this common-sense implication.

quan(?q,?u,?o)&verb(?q,買)&agent(?q,?a)&price(?o,?p)
→quan($q,元,#)=quan(?q,?u,?o)x?p&verb($q,付)&agent($q,?a)

In the above implication inference rule, "quan(?q,?u,?o)&⋯&price(?o,?p)" is the premise of the rule and "quan($q,元,#)=⋯&agent($q,?a)" is the consequence of the rule. Please note that "$q" indicates a unique ID generated by the IE.

爸爸買了 3 本 329 元的故事書和 2 枝 465 元的鋼筆(Papa bought three $329 books and two $465 pens)，爸爸共要付幾元(How much money did Papa pay)?
quan(q1,本,$n1_p$)=3&verb(q1,買)&agent(q1,爸爸)&head($n1_p$,故事書)&price($n1_p$,329) quan(q2,枝,$n2_p$)=2&verb(q2,買)&agent(q2,爸爸)&head($n2_p$,鋼筆)&price($n2_p$,465) ASK Sum(quan(?q,元,#),verb(?q,付)&agent(?q,爸爸))

Figure 3. An example for deriving new facts.

After unifying this inference rule with the facts in Figure 3, we can get two possible bindings (for *q1* and *q2*, respectively). The following shows the binding of *q1*.

quan(q1,本,n1)&verb(q1,買)&agent(q1,爸爸)&price(n1,329)
→quan(q3,元,#)=quan(q1,本,n1)×329&verb(q3,付)&agent(q3,爸爸)

Since "quan(q1,本,n1)×329 = 3×329 = 987", the consequence of the above inference will generate three new facts "quan(q3, 元, #) = 987", "verb(q3, 付)" and "agent(q3, 爸爸)". The semantics of the consequence is "爸爸付 987 元(Papa pays 987 dollars)". Likewise, the consequence of another binding of this inference rule will also generate three new facts "quan(q4, 元, #) = 930", "verb(q4, 付)" and "agent(q4, 爸爸)". By taking these new facts into account, the utility call "Sum(quan(?q,元,#), verb(?q,付)&agent(?q,爸爸))" can thus return the correct answer "1917".

Furthermore, the unification process in a conventional IE is based on string-matching. The expression "qaun(?q, 枝, 筆)" can be unified with a fact "quan(q1, 枝, 筆)". However, it cannot be unified with the fact "quan(q2, 朵, 花)". String-matching guarantees that the IE will not operate on undesired quantities. But, it sometimes prevents the IE from operating on desired quantities. For instance, in Figure 4, two quantity-facts "quan(q1,枝,$n1_p$) = 2361" and "quan(q2,枝,$n2_p$) = 1587" are converted from "2361 枝紅筆(2361 red pens)" and "1587 枝藍筆(1587 blue pens)", respectively. The first argument of "Sum(…)" is "quan(?q, 枝, 筆)" because "幾枝筆(how many pens)" is concerned in the question. The conventional unification is not able to unify "quan(?q, 枝, 筆)" to either "quan(q1, 枝, $n1_p$)" or "quan(q2, 枝, $n2_p$)" due to different strings of the third arguments. However, from the semantic point of view, "quan(?q, 枝, 筆)" should be unified with both "quan(q1, 枝, $n1_p$)" and "quan(q2, 枝, $n2_p$)", because $n1_p$ and $n2_p$ represent "紅筆(red pens)" and "藍筆(blue pens)" respectively (and either one is a kind of "筆(pen)").

文具店進貨2361枝紅筆和1587枝藍筆(A stationer bought 2361 red pens and 1587 blue pens), 文具店共進貨幾枝筆(How many pens did the stationer buy)?
quan(q1,枝,$n1_p$)=2361&verb(q1,進貨)&agent(q1,文具店)&head($n1_p$,筆)&color($n1_p$,紅) quan(q2,枝,$n2_p$)=1587&verb(q2,進貨)&agent(q2,文具店)&head($n2_p$,筆)&color($n2_p$,藍) ASK Sum(quan(?q,枝,筆),verb(?q,進貨)&agent(?q,文具店))

Figure 4. An example for requiring semantic-matching

Therefore, a *semantic matching* method is proposed to be incorporated into the unification procedure. The idea is to match the semantic constituent sets of the two arguments

involved in unification. For example, while matching the third arguments of two functions during unifying the *request*[6] "quan(?q, 枝, 筆)" with the fact "quan(q1, 枝, $n1_p$)", IE will construct and compare two semantic constituent sets, one is for "筆" and the other is for "$n1_p$". Let SCS denote "semantic constituent set" and SCS(*x*) denote the semantic constituent set of *x*. In our approach, "SCS(筆) = {筆}"[7] and "SCS($n1_p$) = {筆, color(紅)}"[8] . Since "SCS(筆)" is covered by the "SCS($n1_p$)", "quan(?q, 枝, 筆)" can be unified with "quan(q1, 枝, $n1_p$)". Likewise, "quan(?q, 枝, 筆)" can be unified with "quan(q2, 枝, $n2_p$)" because "SCS($n2_p$) = {筆, color(藍)}" covers "SCS(筆)". As the result, the utility call "Sum(quan(?q,枝,筆), verb(?q,進貨)&agent(?q,文具店))" will obtain the correct answer "3948". On the other hand, if the question is "文具店共進貨幾枝紅筆(How many red pens did the stationer buy)?", the request will become "quan(?q, 枝, $n3_p$)", where $n3_p$ is a pseudo nonterminal consisting of the terminals "紅(red)" and "筆(pen)" under the noun phrase "幾枝紅筆(how many red pens)". Since "SCS($n3_p$) = {筆, color(紅)}", "quan(?q, 枝, $n3_p$)" can be unified only with "quan(q1, 枝, $n1_p$)". It cannot be unified with "quan(q2, 枝, $n2_p$)" because SCS($n3_p$) cannot be covered by SCS($n2_p$). Therefore, the quantity of "藍筆(blue pens)" will not be taken into account for the question "文具店共進貨幾枝紅筆(How many red pens did the stationer buy)?".

3.4.2 Verb Entailment (Jurafsky & Martin, 2000)

Since we might adopt the verb "買(buy)" in the body text "爸爸買了 3 本 329 元的故事書(Papa bought three $329 books)", but adopt the verb "付(pay)" in the question text "爸爸共要付幾元(How much money did Papa pay)？" (as illustrated in the previous section), we need the knowledge that "buy" implies "pay" to perform logic binding (Moldovan & Rus, 2001). Verb entailment is thus required to identify whether there is an entailment relation between these two verbs (Hashimoto *et al*., 2009). Verb entailment detection is an important function for the IE (de Salvo Braz *et al*., 2006), as it can indicate the event progress and the status changing. In the math problem "Bill had no money. Mom gave Bill two dollars, and Dad gave Bill three dollars. How much money Bill had then?", the entailment between "give (給)" and "have (有)" can update the status of Bill from "no money", then "two dollars", and to the final

---

[6] An FOL predicate/function in an IE utility or in the premise of an inference rule is called a ***request***. A request usually consists of FOL variables.

[7] The SCS of a terminal consists of the terminal string only (e.g., "SCS(筆) = {筆}").

[8] SCS($n1_p$) is constructed by two steps. First, enumerate all facts whose first arguments are $n1_p$. Second, for each enumerated fact, denote the predicate name as Child-Role and the SCS of the second argument as Child-SCS. If Child-Role is "head", put the elements of Child-SCS into SCS($n1_p$). Otherwise, for each string s in Child-SCS, put the string "Child-Role(s)" into SCS($n1_p$). In the first step, the facts "head($n1_p$, 筆)" and "color($n1_p$, 紅)" are picked out. In the second step, the strings "筆" and "color(紅)" are put into SCS($n1_p$).

answer “five dollars”.

We define the verb entailment problem as follows: given an ordered verb pair “(v1, v2)” as input, we want to detect whether the entailment relation ‘v1 → v2’ holds for this pair. E-HowNet (Chen *et al*., 2009; Huang *et al*., 2014) is adopted as the knowledge base for solving this problem. For the previous example verb “give (給)”, we can find its conflation of events, which has been described as the phenomenon involved in predicates where the verb expresses a co-event or accompanying event, rather than the main event (Talmy, 1972; Haugen, 2009; Mateu, 2012), from E-HowNet as shown in Figure 5. The conflations of events are defined by predicates and their arguments (Huang *et al*., 2015), as shown in Figure 5.

Conflation of events:	lose→agent({give|給})=theme({lose|失去}); lose→theme({give|給})=possession({lose|失去}); obtain→theme({give|給})=possession({obtain|得到}); obtain→target({give|給})=theme({obtain|得到}); receive→target({give|給})=agent({receive|收受}); receive→theme({give|給})=possession({receive|收受})

Figure 5. The conflation events of the verb “give (給)”.

Verb entailment is vital for solving the elementary school math problem. Consider the following math problem as a simple example:

> 老師原有 9 枝鉛筆,送給小朋友 5 枝後,老師還有幾枝筆？(The teacher has 9 pencils. After giving his students 5 pencils, how many pencils he has?)

The verbs are “有(have)” and “送給(give as a gift)” in this problem. If we want to derive the concept of “有(have)” from “送給(give as a gift)”, we can follow the direction of their definitions in E-HowNet: “送給(give as a gift)” is a hyponym of “給(give)”, and one of its implication from the conflation of events is “得到(obtain)”, which is a hyponym of “有(have)”.

However, for the four verbs in this derivation, implications are defined only in the verb “給(give)”. As we can see, given all those definitions of words in E-HowNet, we need to find a valid path (which may involve word sense disambiguation) to determine whether there is an entailment between two verbs. Therefore, we need a model to automatically build the relations of these verbs by finding paths from E-HowNet or other resources, and then rank or validate these paths to find the verb entailment. The conflation of events also indicates that when the entailed verb pair is detected, we may further map semantic roles of these two verbs to

proceed the inference and find the solution (Wang & Zhang, 2009).

4. Proposed Statistical Framework

Since the accuracy rate of the Top-1 SR tree cannot be 100%, and the decisions made in the following phases (i.e., STC, LFC and IE) are also uncertain, we need a statistical framework to handle those non-deterministic phenomena. Under this framework, the problem of getting the desired answer for a given WMP can be formulated as follows:

$$\widehat{Ans} = \arg\max_{Ans} \mathrm{P}\left(Ans \middle| Body, Qus \right) \tag{1}$$

Where $\widehat{Ans}$ is the obtained answer, *Ans* denotes a specific possible answer, *Body* denotes the given body text of the problem, and *Qus* denotes the question text of the problem.

The probability factor in the above equation can be further derived as follows via introducing some related intermediate/latent random variables:

$$\begin{aligned}
&\mathrm{P}\left(Ans \middle| Body, Qus \right) \\
&= \sum \mathrm{P}\left(Ans, IR, LF_B, LF_Q, SM_B, SM_Q, ST \middle| Body, Qus \right) \\
&\approx \max \mathrm{P}\left(Ans, IR, LF_B, LF_Q, SM_B, SM_Q, ST \middle| Body, Qus \right) \\
&\approx \max \mathrm{P}\left(Ans \middle| IR, LF_B, LF_Q \right) \times \mathrm{P}\left(IR \middle| LF_B, LF_Q, ST \right) \times \mathrm{P}\left(LF_B \middle| SM_B, ST \right) \\
&\quad \times \mathrm{P}\left(LF_Q \middle| SM_Q, ST \right) \times \mathrm{P}\left(ST \middle| SM_B, SM_Q \right) \times \mathrm{P}\left(SM_B \middle| Body \right) \times \mathrm{P}\left(SM_Q \middle| Qus \right)
\end{aligned} \tag{2}$$

IR : Inference Rules Applied.
LF_B : Logic Form of Body text.
LF_Q : Logic Form of Question text.
SM_B : Semantic Representation of Body text.
SM_Q : Semantic Representation of Question text.
ST : Solution Type.

In the above equation, we will further assume that $\mathrm{P}(Ans|IR,LF_B,LF_Q) \approx \mathrm{P}(Rm)$, where *Rm* is the remaining logic factors in LF_Q after the IE has bound it with LF_B (with referring to the knowledge-base adopted). Last, *Viterbi* decoding (Seshadri & Sundberg, 1994) could be used to search the most likely answer with the above statistical model.

To obtain the associated parameters of the model, we will first get the initial parameter-set from a small seed corpus annotated with various intermediate/latent variables involved in the model. Afterwards, we perform *weakly supervised learning* (Artzi & Zettlemoyer, 2013) on a partially annotated training-set (in which only the answer is annotated with each question). That is, we iteratively conduct beam-search (with the parameter-set obtained from the last iteration) on this partially annotated training-set starting from the given

body text (and question text) to the final obtained answer. If the annotated answer match some of the obtained answers (within the search-beam), simply pick up the matched path with the maximal likelihood value. We then re-estimate the parameter-set (of the current iteration) from those picked up paths. If the annotated answer cannot match any of the obtained answers (within the search-beam), we simply drop that case, and then repeat the above re-estimation procedure.

5. Current Status and Future Work

Currently, we have completed all the associated modules (including Word Segmenter, Syntactic Parser, Semantic Composer, STC, LFC, IE, and Explanation Generation), and have manually annotated 75 samples (from our elementary school math corpus) as the seed corpus (with syntactic tree, semantic tree, logic form, and reasoning chain annotated). Besides, we have cleaned the original elementary school math corpus and encoded it into the appropriate XML format. There are total 23,493 problems from six different grades; and the average number of words of the body text is 18.2 per problem. Table 3 shows the statistics of the converted corpus.

Table 3. MWP corpus statistics and Average length per problem.

Corpus	Num. of problems
Training Set	20,093
Develop Set	1,700
Test Set	1,700
Total	23,493

MWP corpus statistics

Corpus	Avg. Chinese Chars.	Avg. Chinese Words
Body	27	18.2
Question	9.4	6.8

Average length per problem

We have completed a prototype system which is able to solve 11 different solution types (including *Multiplication*, *Summation*, *Subtraction*, *Floor-Division*, *Algebra*, *Comparison*, *Surplus*, *Difference*, *Ceil-Division*, *Common-Division* and *Addition*), and have tested it on the seed corpus. The success of our pilot run has demonstrated the feasibility of the proposed approach. We plan to use the next few months to perform weakly supervised learning, as mentioned above, and fine tune the system.

6. Related Work

To the best of our knowledge, all those MWP solvers proposed before year 2014 adopted the rule-based approach (Mukherjee & Garain, 2008). For example, Bobrow's STUDENT

(Bobrow, 1964; Slagle, 1965) used format matching to map the input English sentence into the corresponding logic statement (all start with predicate "EQUAL"). Another system, WORDPRO, was developed by Fletcher (1985) to understand and solve simple one-step addition and subtraction arithmetic word problems designed for third-grade children. It did not accept the surface representation of text as input. Instead it begins with a set of propositions (manually created) that represent the text's meaning. Afterwards, the problem was solved with a set of rules (also called schemas), which matched the given proposition and then took the corresponding actions. Besides, it adopted key word match to obtain the answer.

Solving the problem with schemata was then adopted in almost every later system (Mukherjee & Garain, 2008). In 1986, ARITHPRO was designed with an inheritance network in which word classes inherit attributes from those classes above them on a verb hierarchy (Dellarosa, 1986). The late development of ROBUST (Bakman, 2007) demonstrated how it could solve free format word problems with multi-step arithmetic through splitting one single sentence into two formula propositions. In this way, transpositions of problem sentences or additional irrelevant data to the problem text do not affect the problem solution. However, it only handles state change scenario. In 2010, Ma *et al*. (Ma *et al*., 2010) proposed a MSWPAS system to simulate people's arithmetic multi-step addition and subtraction word problems behavior. It uses frame-based calculus and means-end analysis (AI planning) to solve the problem with pre-specified rules. In 2012, Liguda and Pfeiffer (Liguda & Pfeiffer, 2012) proposed a model based on augmented semantic networks to represent the mathematical structure behind word problems. It read and solved mathematical text problems from German primary school books. With more attributes associated with the semantic network, it claimed that the system was able to solve multi-step word problems and complex equation systems and was more robust to irrelevant information. Also, it was declared that it was able to solve all classes of problems that could be solved by the schema-based systems, and could solve around 20 other classes of word problems from a school book which were in most cases not solvable by other systems.

Recently, Hosseini *et al*. (2014) proposed a Container-Entity based approach, which solved the math word problem with a state transition sequence. Each state consists of a set of containers, and each container specifies a set of entities identified by a few heuristic rules. How the quantity of each entity type changes depends on the associated verb category. Each time a verb is encountered, it will be classified (via a SVM, which is the only statistical module adopted) into one of the seven categories which pre-specify how to change the states of associated entities. Therefore, logic inference is not adopted. Furthermore, the anaphora and co-reference are left un-resolved, and it only handles addition and subtraction.

Kushman *et al*. (2014) proposed the first statistical approach, which used a few heuristic rules to extract the algebra equation templates (consists of variable slots and number slots)

from a set of problems annotated with equations. For a given problem, all possible variable/number slots are identified first. Afterwards, they are aligned with those templates. The best combination of the template and alignment (scored with a statistical model) is then picked up. Finally, the answer is obtained from those equations instantiated from the selected template. However, without really understanding the problem (i.e., no semantic analysis is performed), the performance that this approach can reach is limited; also, it is sensitive to those irrelevant statements (Hosseini *et al*., 2014). Furthermore, it can only solve algebra related problems. Last, it cannot explain how the answer is obtained.

The most recent statistical approach was proposed by Roy *et al*. (2015), which used 4 cascade statistical classifiers to solve the elementary school math word problems: *quantity identifier* (used to find out the related quantities), *quantity pair classifier* (used to find out the operands), *operation classifier* (used to pick an arithmetic operation), and *order classifier* (used to order operands for subtraction and division cases). It not only shares all the drawbacks associated with Kushman *et al*. (2014), but also limits itself for allowing only one basic arithmetic operation (i.e., among addition, subtraction, multiplication, division) with merely 2 or 3 operand candidates.

Our proposed approach differs from those previous approaches by combining the statistical framework with logic inference. Besides, the tag-based approach adopted for selecting the appropriate information also distinguishes our approach from that of others.

7. Conclusion

A tag-based statistical framework is proposed in this paper to perform understanding and reasoning for solving MWP. It first analyzes the body and question texts into their corresponding semantic trees (with anaphora/ellipse resolved and semantic role labeled), and then converted them into their associated tag-based logic forms. Afterwards, the inference (based on the question logic form) is performed on the logic facts derived from the body text. The combination of the statistical frame and logic inference distinguishes the proposed approach from other approaches. Comparing to those rule-based approaches, the proposed statistical approach alleviates the ambiguity resolution problem; also, our tag-based approach provides the flexibility of handling various kinds of related questions with the same body logic form. On the other hand, comparing to those purely statistical approaches, the proposed approach is more robust to the irrelevant information and could more accurately provide the answer.

The contributions of our work mainly lie in: (1) proposing a tag-based logic representation which makes the system less sensitive to the irrelevant information and could provide answer more precisely; (2) proposing a statistical framework for performing reasoning from the given text.

Acknowledgment

We would like to thank Prof. Wen-Lian Hsu for suggesting this research topic and making the original elementary school math corpus available to us, and Prof. Keh-Jiann Chen for providing the resources and supporting this project. Besides, our thanks should be extended to Dr. Yu-Ming Hsieh and Dr. Ming-Hong Bai for implementing the syntactic parser and the semantic composer, respectively. Also, we would like to thank Prof. Chin-Hui Lee for suggesting the solution type. Last, our thanks should also go to Ms. Su-Chu Lin for manually annotating the corpus.

References

Allen, J. F. (2014). Learning a Lexicon for Broad-Coverage Semantic Parsing. In the *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, 1-6.

Artzi, Y., & Zettlemoyer, L. (2013). Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, *1*(2013), 49-62.

Bakman, Y. (2007). *Robust Understanding of Word Problems With Extraneous Information*. Retrieved from arXiv:math/0701393.

Ballard, B. & Biermann, A. (1979). PROGRAMMING IN NATURAL LANGUAGE : "NLC" AS A PROTOTYPE. *ACM-Webinar*, 1979, DOI: 10.1145/800177.810072.

Berant, J., Chou, A., Frostig, R., & Liang, P. (2013). Semantic Parsing on Freebase from Question-Answer Pairs. *Conference on Empirical Methods in Natural Language Processing (EMNLP)2013*, 1533-1544.

Biermann, A. W., & Ballard, B. W. (1980). Toward Natural Language Computation. *American Journal of Computational Linguistic*, *6*(2), 71-86.

Biermann, A., Rodman, R., Ballard, B., Betancourt, T., Bilbro, G., Deas, H., Fineman, L., Fink, P., Gilbert, K., Gregory, D., & Heidlage, F. (1982). INTERACTIVE NATURAL LANGUAGE PROBLEM SOLVING:A PRAGMATIC APPROACH. In *Proc. of the first conference on applied natural language processing*, 180-191.

Bobrow, D. G. (1964). *Natural language input for a computer problem solving system*. Ph.D. Dissertation, Massachusetts Institute of Technology.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3). Doi:10.1145/1961189.1961199.

Charniak, E. (1968). *CARPS, a program which solves calculus word problems*. Report MAC-TR-51, Project MAC, MIT.

Charniak, E. (1969). Computer solution of calculus word problems. In *IJCAI'69 Proc. of International Joint Conference on Artificial Intelligence*, 303-316.

Chen, K.-J., Huang, S.-L., Shih, Y.-Y., & Chen, Y.-J. (2005). Extended-HowNet- A Representational Framework for Concepts. *OntoLex 2005 - Ontologies and Lexical Resources IJCNLP-05 Workshop*, Jeju Island, South Korea.

Chen, K.J., & Ma, W.Y. (2002). Unknown Word Extraction for Chinese Documents. In *Proceedings of Coling 2002*, 169-175.

Das, D., Chen, D., Martins, A. F. T., Schneider, N., & Smith, N. A. (2014). Frame-Semantic Parsing. *Computational Linguistics*, *40*(1), 9-56.

Dellarosa, D. (1986). A computer simulation of children's arithmetic word-problem solving. *Behavior Research Methods, Instruments, & Computers*, *18*(2), 147-154.

Fletcher, C. R. (1985). COMPUTER SIMULATION -- Understanding and solving arithmetic word problems: A computer simulation. *Behavior Research Methods, Instruments, & Computers*, *17*(5,) 565-571.

Gelb, J. P. (1971). Experiments with a natural language problem solving system. In *Proc. of IJCAI-71*, 455-462.

Hashimoto, C., Torisawa, K., Kuroda, K., De Saeger, S., Murata, M., & Kazama, J. J. (2009). Large-scale verb entailment acquisition from the web. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, *3*, 1172-1181.

Haugen, J. D. (2009). Hyponymous objects and Late Insertion. *Lingua*, 119, 242-262.

Hosseini, M. J., Hajishirzi, H., Etzioni, O., & Kushman, N. (2014). Learning to Solve Arithmetic Word Problems with Verb Categorization. *EMNLP(2014)*, 523-533.

Hsieh, Y.-M., Chang, J. S., & Chen, K.-J. (2014). Ambiguity Resolution for Vt-N Structures in Chinese. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 928-937.

Hsieh, Y.-M., Lin, S.-C., Chang, J. S., & Chen, K.-J. (2013). Improving Chinese Parsing with Special-Case Probability Re-estimation. In *Proceedings of 2013 International Conference on Asian Language Processing (IALP)*, 177-180.

Hsieh, Y.-M., Yang, D.-C., & Chen, K.-J. (2007). Improve Parsing Performance by Self-Learning. *International Journal of Computational Linguistics and Chinese Language Processing*, *12*(2), 195-216.

Huang, S.-L., Hsieh, Y.-M., Lin, S.-C., & Chen, K.-J. (2014). Resolving the Representational Problems of Polarity and Interaction between Process and State Verbs. *International Journal of Computational Linguistics and Chinese Language Processing (IJCLCLP)*, *19*(2), 33-52.

Huang, S.-L., Lin, S.-C., Ma, W.-Y., & Chen, K.-J. (2015). *Semantic Roles and Semantic Role Labeling.* (CKIP technical report no. 2015-01). Institute of Information Science, Academia Sinica.

Huang, C. T., Lin, Y. C., & Su, K. Y. (2015). Explanation Generation for a Math Word Problem Solver. *International Journal of Computational Linguistics and Chinese Language Processing (IJCLCLP)*, *20*(2), 27-44.

Huang, Z., Thint, M., & Qin, Z.(2008). Question classification using head words and their hypernyms. In *Proceeding of EMNLP '08 Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 927-936.

Jurafsky, D., & Martin, J. H. (2000). *Speech and Language Processing*. New Jersey: Prentice Hall.

Jurcıcek, F., Mairesse, F., Gašic, M., Keizer, S., Thomson, B., Yu, K., Young, S., & Gasic, M. (2009). Transformation-based Learning for Semantic parsing. In *Proceedings of INTERSPEECH 2009*, 2719-2722.

Kushman, N., Artzi, Y., Zettlemoyer, L., & Barzilay, R. (2014). Learning to Automatically Solve Algebra Word Problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 271-281.

Liguda, C., & Pfeiffer, T. (2012). Modeling math word problems with augmented semantic networks. *NLDB 2012*, 247-252.

Loni, B. (2011). A survey of State-of-the-Art Methods on Question Classification. *Literature Survey*, Published on TU Delft Repository, 2011 Jun.

Ma, W.-Y., & Chen, K.-J. (2003). Introduction to CKIP Chinese Word Segmentation System for the First International Chinese Word Segmentation Bakeoff. In *Proceedings of ACL, Second SIGHAN Workshop on Chinese Language Processing*, 168-171.

Ma, Y. H., Zhou, Y., Cui, G. Z., Ren, Y., & Huang, R. H. (2010). Frame-Based Calculus of solving Arithmetic MultiStep Addition and Subtraction word problems. In *2010 Second International Workshop on Education Technology and Computer Science*, 476-479.

Mateu, J. (2012). *Conflation and incorporation processes in resultative constructions*. In Violeta Demonte & Louise McNally (eds.), Telicity, Change, and State: A Cross-Categorial View of Event Structure, Oxford: Oxford University Press, 252-278.

Moldovan, D., & Rus, V. (2001). Logic Form Transformation of WordNet and Its Applicability to Question Answering. In *ACL '01 Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, 402-409.

Mukherjee, A., & Garain, U. (2008). A review of methods for automatic understanding of natural language mathematical problems. *Artif Intell Rev*, *29*(2), 93-122.

Roy, S. I., Vieira, T. J. H., & Roth, D. I.(2015). Reasoning about Quantities in Natural Language. *TACL*, *3*, 1-13.

Russell, S. J. & Norvig, P. (2009). *Artificial Intelligence : A Modern Approach*(3rd Edition), Prentice Hall.

de Salvo Braz, R., Girju, R., Punyakanok, V., Roth, D., & Sammons, M. (2006). An inference model for semantic entailment in natural language. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment.*, Springer Berlin Heidelberg, 2006, 261-286.

Seshadri, N., Sundberg, C.-E.W. (1994). List Viterbi Decoding Algorithms with Applications. *IEEE Transactions on Communications*, *42*(234), 313-323.

Slagle, J. R. (1965). Experiments with a deductive question-answering program. *J-CACM*, *8*(12), 792-798.

Strassel, S., Adams, D., Goldberg, H., Herr, J., Keesing, R., Oblinger, D., Simpson, H., Schrag, R., & Wright, J. (2010). The DARPA Machine Reading Program - Encouraging Linguistic and Reasoning Research with a Series of Reading Tasks. *LREC 2010*.

Talmy, L. (1972). *Semantic Structures in English and Atsugewi*. PhD thesis, Berkeley: University of California at Berkeley.

Tsai, Y.-F., & Chen, K.-J. (2004). Reliable and Cost-Effective Pos-Tagging. *International Journal of Computational Linguistics & Chinese Language Processing*, *9*(1), 83-96.

Tseng, H. H., & Chen, K.-J. (2002). Design of Chinese Morphological Analyzer. In *Proceedings of SIGHAN 2002*, 49-55.

Wang, R., & Zhang, Y. (2009). Recognizing textual relatedness with predicate-argument structures. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, *2*, 784-792.