Arithmetic Word Problem Solver

CSE 537: Artificial Intelligence Course Project

Janice Darling (111887280), Rohan Vaish (111447435), Harsha Chandwani (111481387)

Team - ProblemBusters

Abstract

This paper seeks to explore the use of different natural language processing approaches to build a model for automatically solving elementary arithmetic word problems. The task falls under the field of Question Answering, which is a subfield of Information Retrieval and Natural Language Processing. As such, the steps to returning an answer to a question will involve the simplification of the input text if necessary. This serves to facilitate the extraction of relevant information such as subjects, objects and quantities from the input text. The extraction step is then followed by the processing of the relevant information to arrive at an answer.

1 Task Definition

The previous attempts by other individuals at implementing programs to solve arithmetic problems illustrate the interest and challenge posed by this Artificial Intelligence subject. Simple arithmetic problems are often modelled similarly. That is, these problems can oftentimes be represented by equations in which there is one value unknown. Here, we seek to tackle the problem of implementing a system that accepts as input a piece of text. This text will model a rudimentary word problem that requires the addition or subtraction of numerical quantities contained within the text and return a result that is relevant to the question asked. For example, given the text "John has 3 apples . He gave 1 to Sally . How many apples does John possess ?" as input, the system should be able to identify or recognize provide some representations of the entities owners and their quantities. After which, the necessary operation or operations should be applied to the quantities and a correct solution of 2 should output.

For this work, we will be seeking to replicate the model as described by Sundaram and Khemani as well as to consider elements presented by Hosseini et al. such as the categorization of verbs (Hosseini et al. 2014). That is, in addition to simplifying the code to allow for generalization to questions containing more than two entities, our model will seek to solve questions made up of a variety of structures. This will be done by investigating how the constituency and dependency parses may be helpful in returning a correct answer to the question. Also, an user interface will be added to allow added interactivity with the implementation.

2 Motivation

Question Answering comprises elements of Information Retrieval and Natural language Processing (NLP). Though the discipline has grown to include a variety of ways to accomplish the task of providing a correct answer to a textual input, the general approach in previous implementations of question answering systems is to query a database which may either be specifically related to the question (closed domain) or a general corpus of natural language documents. Arithmetic Problem Solving indirectly builds from this general approach. The problem is presented in a succinct manner. Therefore, these problems are attractive to NLP especially with the availability of different tools and APIs such as the StanfordCoreNLP package, we are able to focus on the linguistic structure of the problems as well as investigating how the structure may be helpful in returning a correct answer for the arithmetic problem. Here an opportunity has presented itself to learn about the field Question Answering as well as NLP topics such as part-of-speech tagging and dependency parsing.

3 Description and Tasks Accomplished

The system pipeline is illustrated in Figure 1 and takes as input an arithmetic problem in the text form.

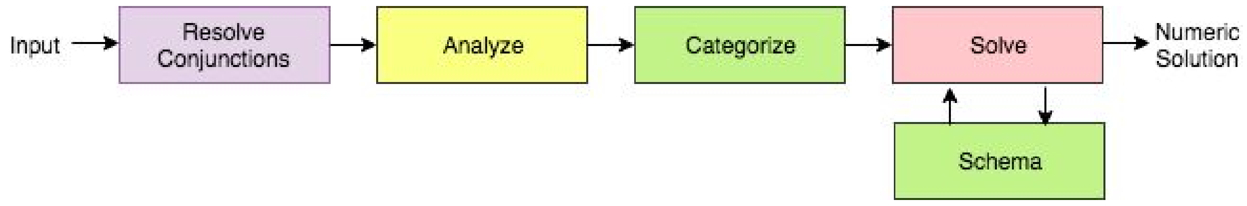


Fig 1: High-level overview of Arithmetic Solver system

The first step in the pipeline is to simplify the input if necessary. This simplification is necessary to begin the process of knowledge representation and when the original problem that was input to the system contains conjunctions joining two simple sentences into a compound one, or the conjunction introduces ambiguity (Shi et al. 2015). For example, given the problem “Harry had 2 bikes and Jack had 4. How many bikes do they have in total ?”, the systems parser would identify the conjunction “and” and would resolve this by first splitting the sentence on the conjunction. The left and right segments are now “Harry had 2 bikes” and

"Jack had 4" respectively. The next step is to identify the verbs in both segments using the part of speech tag. The part of speech tag serves to identify what role a word plays. Here, in searching for a verb, any part of speech (POS) tag beginning with 'VB' that was returned by the StanfordCoreNLP pos-tagger was considered a verb. Later, the full tag is considered when capturing the tense of the verbs. If the right segment is missing a verb, then that verb is copied over from the left segment (Sundaram and Khemani 2015). The next step is to check for prepositional phrases. Prepositional phrases have a structure of PP → P (NP) where P is a preposition and NP is a noun phrase which in turn has the structure NP → (DET) (ADJ) N (PP) where DET is determiner, ADJ is adjective, N is noun, and PP as defined above. If the one segment contains a prepositional phrase and the other does not, then the prepositional phrase will be copied to the empty one. After this resolution is completed, the sentences are joined are returned as the word problem.

In the analyze phase, the pronouns are resolved. Here, any personal pronouns such as he or she is replaced by the first owner that is encountered. This simplifies a problem with sentences like "Harry had 2 bikes and he gave 1 to Ben". Also in the analyze phase, the dependency parse of the sentences are used to provide the syntactic structure. For example the sentence "Harry had 2 bikes" produces the following dependency structure: (u'ROOT', 0, 2), (u'nsubj', 2, 1), (u'nummod', 4, 3), (u'dobj', 2, 4), where the first element in the tuple is the dependency, the second is the source node and the third is the target node. Therefore, "had" is the root and "Harry" is the subject of "had". The dependencies were then sorted by source which allowed for easier identification of the subjects in addition to using the POS tag and searching for "NNP" indicating proper noun if a nominal modifier is encountered. Quantities are collected by checking if the POS tag is "CD" indicating a cardinal number and the objects are extracted by taking the word after the after the number. Finally, verbs are collected by checking the POS tag and also recording the tense of the verb.

After analyzing, the task of categorizing is accomplished. Here, entity objects with member variables value and owner are created from the information returned from analyzing. The values of the relevant entities are updated in the solve step. Here, the verb returned by analyze is used to return the necessary computation as described below in section 6. Finally, the subject in the question is used to make a decision on which entity object value to return. Therefore to summarize, the main accomplishments here were the conjunction resolution, entity extraction, and question categorization.

4 Evaluation

The only evaluation metric that will be considered is **accuracy**. This will be based on the percentage of questions that are answered correctly against a compiled list. Our implementation was checked against the baseline model as well as the model proposed in Learning to Solve Arithmetic Word Problems with Verb Categorization (Hosseini et al. 2014).

A characteristic of question answering systems is that they often require the input to be structured in a specific manner in order to be able to extract information for the task of returning an answer. With this in mind, we have chosen not to evaluate the accuracy of our system based on the data sets used by others as our approach focuses heavily on the dependency parse and part of speech tagger outputs for each sentence in the question.

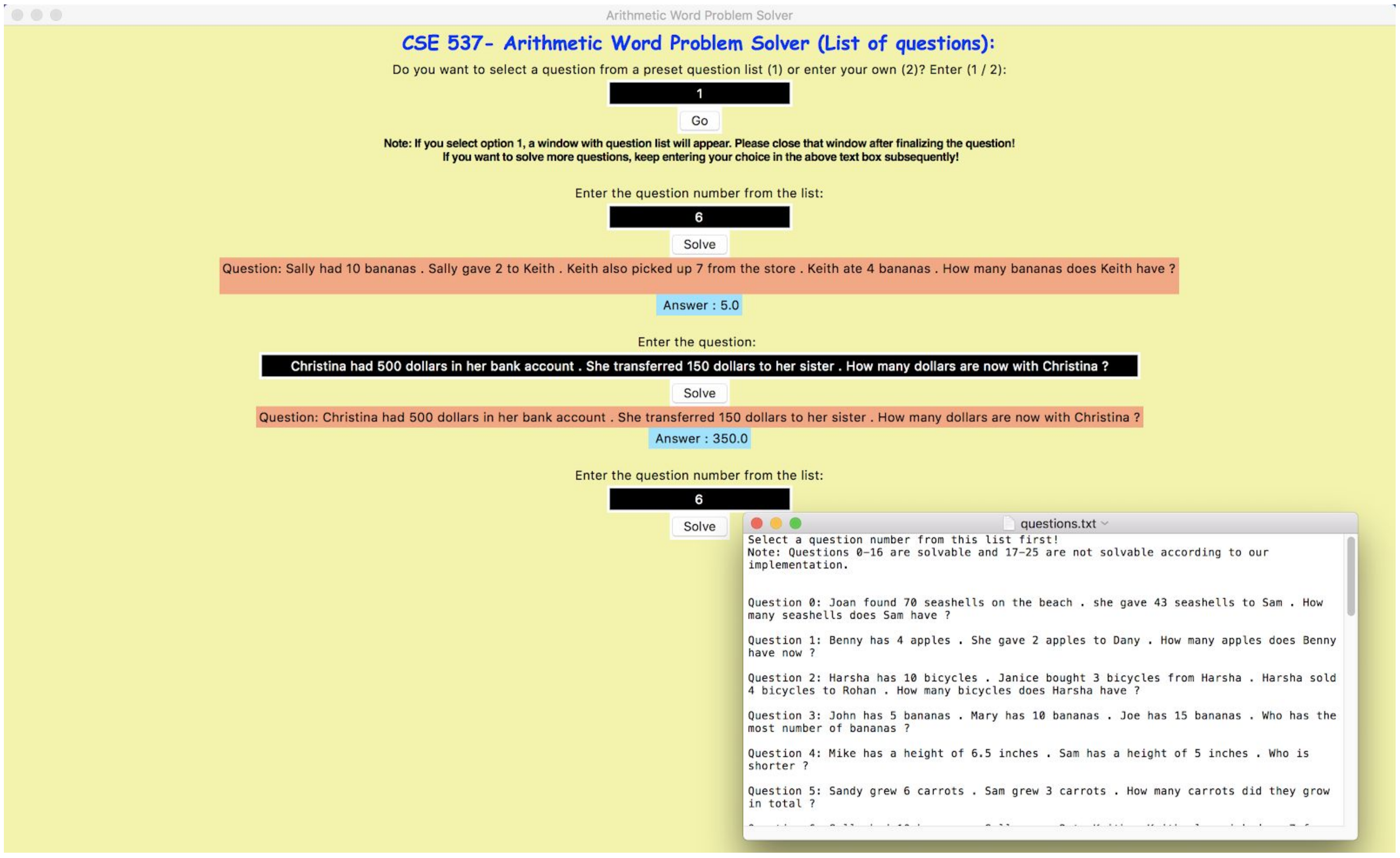
5 Results

We have evaluated our model on a compiled list of 30 questions, out of which 20 were solvable and 10 were not. Based on this, the accuracy of the model comes to **66.66%.** This is compared to the results of the model proposed by Sundaram and Khemani which yielded an accuracy of 88.64% averaged over three datasets, and 77.7% for the model proposed by Hosseini et al. Also, although the accuracy returned by the model is not comparatively high, the idea of focusing on the dependency parse structure of the questions is one that is compendable in terms of generalizing the code to be able to handle a variety of questions. However, in order to precisely evaluate our model's performance against the others discussed here, it becomes necessary to test on the same collection of questions.

Instructions to run the project:

1. Python 2.x is required
2. Download **StanfordCoreNLP** library from https://stanfordnlp.github.io/CoreNLP/ and point the path variable in **config.py** to the downloaded folder.
3. Run the command "**pip install stanfordcorenlp**"
4. Run the command "**sudo python main.py"** from the code directory
5. Window will appear which will contain the instructions on how to proceed further.

Here is a screenshot of the working project implementaion:



CSE 537- Arithmetic Word Problem Solver (List of questions):

Do you want to select a question from a preset question list (1) or enter your own (2)? Enter (1 / 2):

1

Note: If you select option 1, a window with question list will appear. Please close that window after finalizing the question!
If you want to solve more questions, keep entering your choice in the above text box subsequently!

Enter the question number from the list:

6

Question: Sally had 10 bananas . Sally gave 2 to Keith . Keith also picked up 7 from the store . Keith ate 4 bananas . How many bananas does Keith have ?

Answer : 5.0

Enter the question:

Christina had 500 dollars in her bank account . She transferred 150 dollars to her sister . How many dollars are now with Christina ?

Question: Christina had 500 dollars in her bank account . She transferred 150 dollars to her sister . How many dollars are now with Christina ?

Answer : 350.0

Enter the question number from the list:

6

```
Select a question number from this list first!
Note: Questions 0-16 are solvable and 17-25 are not solvable according to our
implementation.


Question 0: Joan found 70 seashells on the beach . she gave 43 seashells to Sam . How
many seashells does Sam have ?

Question 1: Benny has 4 apples . She gave 2 apples to Dany . How many apples does Benny
have now ?

Question 2: Harsha has 10 bicycles . Janice bought 3 bicycles from Harsha . Harsha sold
4 bicycles to Rohan . How many bicycles does Harsha have ?

Question 3: John has 5 bananas . Mary has 10 bananas . Joe has 15 bananas . Who has the
most number of bananas ?

Question 4: Mike has a height of 6.5 inches . Sam has a height of 5 inches . Who is
shorter ?

Question 5: Sandy grew 6 carrots . Sam grew 3 carrots . How many carrots did they grow
in total ?
```

Fig 2: Screenshot of running project

6 Analysis

The word problem is processed one sentence at a time. For every sentence except the last, we follow the steps as mentioned below.

1. Every sentence is parsed to resolve the conjunctions, pronouns, owners, the verb, the entity and its value.
2. The verb is processed to get the type and the operation corresponding to it. If the type is **unary**, it affects the quantity with just one owner, if it is binary, the quantities with two owners are affected. The effect of the verb on the quantity with an owner is decided by the operation associated with a verb. Unary verbs can have operations like **increase** or **decrease**. Binary verbs can have operations like **LtoR** i.e. subtract

the value from the quantity with the left owner and add the value to the quantity with the right owner or the operation could **RtoL**, which works in the opposite manner.

Once the sentences are processed in this way, we have the final quantities associated with each owner in the problem. So, for the last sentence or the question part of the word problem, we extract the owner from the last sentence and return the final quantity that the owner has.

Below are a few examples of the problems that could be solved by our model:

1. Sally had 10 bananas . Sally gave 2 to Keith . Keith also picked up 7 from the store . Keith ate 4 bananas . How many bananas does Keith have ?
2. Benny has 4 apples . She gave 2 apples to Dany . How many apples does Benny have now ?
3. Mike has a height of 6.5 inches . Sam has a height of 5 inches . Who is shorter ?
4. Sandy grew 6 carrots . Sam grew 3 carrots . How many carrots did they grow in total ?

For a problem to be solvable by our proposed model, following assumptions should be met:

1. The problem should involve just one entity.
2. The model replaces all the pronouns by the first owner of the first sentence of the problem. This may or may not give the correct answer.
3. The operations are performed based on the verbs present in each sentence. Hence, for a successful operation, every sentence should have a verb.
4. The verb should also be present in the schema of the verbs maintained which specifies the type and the operation associated with a verb.
5. The model supports only addition and subtraction.
6. There has to be a mention of some entity or a specified set of keywords in the last statement of the problem in order to compute the correct result.

A few examples of the word problems that could not be solved are:

1. Sara has 31 red and 15 green balloons . Sandy has 24 red balloons . How many red balloons do they have in total ?

   Constraint violated : There are more than one entity i.e. Red and green balloons
2. Jenny ran 0.6 mile and walked 0.4 mile . How much farther did Jenny run than walk ?

   Constraint violated : The verb is not a part of the Verb schema of the model
3. Blake filled a bucket with 0.8 gallon of water . Later , he poured out 0.2 gallon of the water . How

much water is in the bucket ?

Constraint violated : Last statement does not mention the entity or any of the expected keywords.

The image below in figure 2 illustrates the steps in computing a result for the word problem : "Joan found 70 seashells on the beach. She gave 40 seashells to Sam. How many seashells does Joan have now? Answer = 30

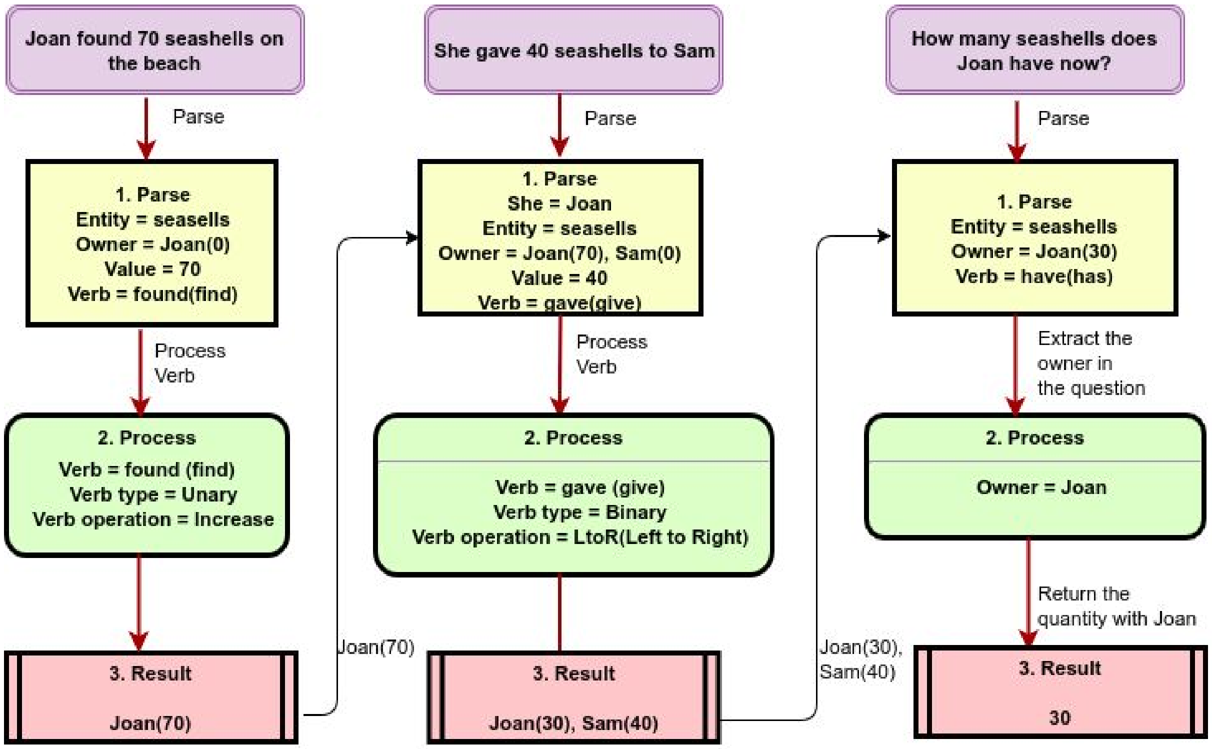


Fig 3: Steps in solving a word problem

7 Conclusion

In conclusion, we sought to build an arithmetic word problem solver that would simplify the system proposed by Sundaram and Khemani which focuses heavily on extracting entities, quantities and owners given that the input was specifically structured. We accomplished this by focusing on the dependency parse tree of the input. The dependency parse allows for the syntactic and semantic information in sentences. As such, it can be as a method to reduce ambiguity when representing the knowledge conveyed in the problem. Based on the dependency parse structure, the problem was searched accordingly for patterns that would result in the correct extraction of quantities, owners and the objects in questions. Therefore, this required creating a new set of problems on which to evaluate the performance.

In order to improve the performance of the system, the part of speech tags of the verbs in question should be utilized. This is as the tense of the verb denoting an alteration in the different quantities involved. For the sake of specificity, consider the example: “John has 4 beers and Peter has 7. Earlier, John gave Peter 2 beers. How many beers did the John originally have ?”. Here the tense of the verb in the sentences defining the environment the environment is the present tense. However, there is information that indicates an earlier change in the quantities in addition to the tense of the question prompting for this change. As such, the tense of the verbs should be taken into consideration (Sundaram and Khemani 2015). In addition to this, the nltk.wordnet package can be used to search for synonyms of verbs in questions that are not defined in the schema necessary for indicating which computation should take place. Finally a deep neural network model can be trained to predict the type of operation to be performed based on the input of the question. This would therefore become a supervised machine learning task and as such require large datasets consisting of arithmetic word problems as well as the labels indicating the computation to be performed.

8 References

Özateş , Şaziye Betül , Arzucan Özgür , and Dragomir Radev. 2016. "Sentence Similarity based on Dependency Tree Kernels for Multi-document Summarization." *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016).* Portorož: European Language Resources Association (ELRA). 23-28.

Hosseini, Mohammad Javad, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. "Learning to Solve Arithmetic Word Problems with Verb Categorization." *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Doha: Association for Computational Linguistics. 523-533.

Mehta, Purvanshi, Pruthwik Mishra, Vinayak Athavale, Manish Shrivastava , and Dipti Misra Sharma. 2017. "The 8th International Joint Conference on Natural Language Processing ." *The Companion Volume of the IJCNLP 2017 Proceedings: System Demonstrations.* Taipei: Asian Federation of Natural Language Processing. 65-68.

Sundaram , Sowmya S, and Deepak Khemani. 2015. "Natural Language Processing for Solving Simple Word Problems." *12th International Conference on Natural Language Processing Proceedings.* Trivandrum: Association for Computational Linguistics.