NLP for Solving Arithmetic Word Problems

M.Uday Theja CS14B044
S.Sai Teja Reddy CS14B051

Introduction

We implement a system which can solve simple arithmetic word problems involving addition and subtraction. The system is given an arithmetic word problem in natural language as input. A temporal state representation is extracted from the question and equations are built from it. These equations are solved and the answer is displayed.

For example:

Input: Mike had 34 peaches at his roadside fruit dish. He went to the orchard and picked peaches to stock up.There are now 86 peaches. How many did he pick?

Output: 52

Arithmetic Problem Representation

- **Entities :** The object in the problem text whose quantity is being observed or is changing throughout the problem.
- **Attributes :** The adjectives or noun modifiers of an entity which differentiate between entities.
- **Containers :** The mention in the text possessing the entity or a location containing the entity.
- **Quantities :** Every container-entity pair is associated with a numerical quantity. This quantity can either be a known number or a variable or an arithmetic expression
- **State :** The world state is represented as a tuple <E,C,R> consisting of Entities E, Containers C and Relations R.

"Joan found 70 blue seashells and 30 red seashells on the beach . "

"There are 22 walnut trees currently in the park ."

"Liz had 9 black kittens. She gave some of her kittens to Joan"

Preprocessing Sentences

Preprocessing sentences involves three steps:
Resolving Conjunctions: Sentences containing *and,but,if* and *then* are split into constituent sentences,

- The sentence is split at the conjunction into two parts Part1 and Part2
- Each part is assumed to have four subparts - preverb(PV), verb(V), afterverb(AV) and preposition phrase(PP).
- Any subpart from Part2 which is empty is populated with the corresponding subpart from Part1 and vice versa.
- The four subparts are concatenated into a sentence. The original sentence is replaced by the two new sentences.

Example:
There are 11 rulers and 34 crayons in the drawer

'Part1': "There are 11 rulers"
'PV': "There" 'V': "are"
'AV': "11 rulers" 'PP': " "
'Part2': "34 crayons in the drawer"
'PV': " " 'V': " "
'AV': "34 crayons" 'PP': "in the drawer"

'Part1': "There are 11 rulers"
'PV': "There" 'V': "are"
'AV': "11 rulers" 'PP': "in the drawer"
'Part2': "34 crayons in the drawer"
'PV': "There" 'V': "are"
'AV': "34 crayons" 'PP': "in the drawer"

Preprocessing Sentences

Preprocessing Currency: Whenever we see text of the type $300 we replace it with 300 dollars. This is done so that the dependency parser correctly parses the sentence.

Coreference Resolution: This step is used to replace pronouns with their referring entities.

Example for coreference resolution:

“Joan found 70 seashells on the beach. She gave few seashells to Sam. “
=>
“ Joan found 70 seashells on the beach. Joan gave few seashells to Sam.”

Preprocessing Example

Input:

"Joan found 70 blue seashells and 30 red seashells on the beach. She gave few blue seashells to Sam. She has 27 blue seashell left. How many blue seashells did she give to Sam ?"

Output:

"Joan found 70 blue seashells on the beach. Joan found 30 red seashells on the beach . Joan gave few blue seashells to Sam. Joan has 27 blue seashell left . How many blue seashells did Joan give to Sam?"

Verb Categorization

We notice that the verb performs some kind of transaction between the two containers in terms of the entity. Based on the transaction, we identify 7 categories for verbs,

OBSERVATION	Quantity of entity is initialized	$N(c_{t1}, e_t) = num_t$	Sam **had** 100 dollars
POSITIVE	Quantity of entity is only increased in the first container	$N(c_{t1}, e_t) \mathrel{+}= num_t$	Sam **found** 100 dollars
NEGATIVE	Quantity of entity is only decreased in the first container	$N(c_{t1}, e_t) \mathrel{-}= num_t$	Sam **lost** 100 dollars
POSITIVE TRANSFER	Quantity of entity is transferred from second container to first container	$N(c_{t1}, e_t) \mathrel{+}= num_t$ $N(c_{t2}, e_t) \mathrel{-}= num_t$	Sam **borrowed** 100 dollars from Jack
NEGATIVE TRANSFER	Quantity of entity is transferred from first container to second	$N(c_{t1}, e_t) \mathrel{-}= num_t$ $N(c_{t2}, et) \mathrel{+}= num_t$	Sam **gave** 100 dollars to Jack
CONSTRUCT	Quantity of entity is increased in both containers	$N(c_{t1}, e_t) \mathrel{+}= num_t$ $N(c_{t2}, e_t) \mathrel{+}= num_t$	The farmer **produced** 100kg rice for the village
DESTROY	Quantity of entity is decreased in both containers	$N(c_{t1}, e_t) \mathrel{-}= num_t$ $N(c_{t2}, e_t) \mathrel{-}= num_t$	The gardener **cut** the weeds in the park

Extracting state representation

After preprocessing the text, we obtain a sequence of sentence fragments $<w_1, ..., w_T, w_x>$. Every w_t has a an entity e_t, its attribute a_t, quantity num_t, verb v_t and at most two containers c_{t1} and c_{t2}.

Identifying entities and containers :

- Extract all numbers and noun phrases (NP).
- h ← set of all nouns which are dependent on a number.
- e_t ← all NP which are headed by a noun in h.
- num_t ← the dependent number of e_t if exists. Bare numbers which are present in numbers and not associated with any e_t yet are associated with the nearest entity in the text. All other num_t are set to unknown.
- a_t ← adjective and noun modifiers of e_t. If not found, are assigned the previously observed attribute for that e_t.
- v_t ← the verb with the shortest path to e_t in the dependency parse tree.
- c_{t1}, c_{t2} ← the containers for entity e_t. If not found, set to previous values by circumscription assumption

Verb Categorization: For each fragment w_t, the verb v_t is categorized into one of the 7 possible categories

Preprocessing Step Output :

“Joan found 70 blue seashells on the beach. Joan found 30 red seashells on the beach . Joan gave few blue seashells to Sam. Joan has 27 blue seashell left . How many blue seashells did Joan give to Sam ?”

Fragmentation Output :

Fragments	h	e_t	a_t	num_t	c_{t1}	c_{t2}	v_t	Verb Category
Joan found 70 blue seashells on the beach.	seashell	70 blue seashells	blue	70	Joan	beach	found	POSITIVE
Joan found 30 red seashells on the beach .	seashell	30 red seashells	red	30	Joan	beach	found	POSITIVE
Joan gave few blue seashells to Sam.	seashell	few blue seashells	blue	$	Joan	Sam	gave	NEGATIVE TRANSFER
Joan has 27 blue seashell left .	seashell	27 blue seashell	blue	27	Joan	Joan	has	OBSERVATION
How many blue seashells did Joan give to Sam ?	seashell	How many blue seashells	blue	$	Joan	Sam	give	NEGATIVE TRANSFER

State Progression :

For each fragment in $<w_1, ..., w_T>$, we build corresponding states $<s_1, ..., s_T>$ and $s_0 \leftarrow$ null. For building state s_{t+1}, we first copy all the contents of state s_t and then create or append a new container-entity if previously not present , of the form **$\{c_t : [\{e_t : , a_t : , num_t : \}]\}$** else if it is already present then we update the existing contents num_t using the verb category of v_t , for container-entity pairs (c_{t1}, e_t) and (c_{t2}, e_t) and quantity num_t We only consider entities asked in w_x to build the states.

Fragments	**Verb Category**	**Relevant States**
Joan found 70 blue seashells on the beach.	POSITIVE	'joan': [{'A': 'blue', 'N': 'J0+70', 'E': 'seashell'}]
Joan found 30 red seashells on the beach .	POSITIVE	'joan': [{'A': 'blue', 'N': 'J0+70', 'E': 'seashell'}]
Joan gave few blue seashells to Sam.	NEGATIVE TRANSFER	'joan': [{'A': 'blue', 'N': 'J0+70-$0', 'E': 'seashell'}] 'sam': [{'A': 'blue', 'N': 'J00+$0', 'E': 'seashell'}]
Joan has 27 blue seashell left .	OBSERVATION	'joan': [{'E': 'seashell', 'N': '27', 'A': 'blue'}], 'sam': [{'E': 'seashell', 'N': 'J00+$0', 'A': 'blue'}]

Building and Solving Equations

An equation is formed when num_t of state s_i consists of an unknown and num_t of state s_{i+1} is a known number and all such equations are solved to find the values of the unknowns.

From previous example, **['J0+70-$0=27'] => {'J0': '0' , '$0': '43'}**

Once the equations are solved and the values of unknowns are used, we need to pick the variables which contribute to the final answer as follows :

- If the verb category of v_x is **OBSERVATION** and w_x contains "begin" or "start" then the answer is the num_t of the first occurrence of e_t in the states.
- Else we attempt to pick the fragments whose verb v_t is similar to v_x and has the same container pairs and add the variable introduced in that fragment to our answer.
- Else we attempt to pick the fragments whose verb v_t is opposite to v_x and has the opposite container pairs and add the variable introduced in that fragment to our answer.
- Else, finally, we pick the final value of the entity e_x in the final state s_T and assign its quantity num_t to the answer

From previous example, **Answer = $0 = 43.**

Results and Evaluation

The performance of our system is evaluated on the dataset of standard primary school test questions. provided by (Hosseini et al., 2014). The performance is compared against two other systems. Thay are WolframAlpha's Full Results API and Roy Roth Illinois' interface. Wolfram Alpha's system expects simple and straightforward queries. Hence, this system can be considered as a Baseline for evaluation.

Roy-Roth's Interface	**95.03%**
Our System	**46.81%**
Wolfram Alpha's API	**4.25%**

Error Analysis

Resolving Conjunctions: Consider the sentence, "Jason has 43 blue and 16 red marbles". The output of our preprocessing step gives "Jason has 43 blue. Jason has 16 red marbles." and so the first fragment is missing the entity "marble". Our algorithm expects each fragment to have an entity and so fails in this case.

Irrelevant Sentences: Consider the problem text, "Joan decided to sell all of her old books . She gathered up 33 books to sell . She sold 26 books in a yard sale . How many books does Joan now have ?". Here "Joan decided to sell all of her old books ." is irrelevant. But the system categorizes “sell” as Negative Transfer and creates a state for it. This leads to an incorrect final answer.

World Knowledge: Consider the sentence, "Sara ’s high school played 12 basketball games this year. The team won most of their games .". Here, "Sara’s high school" is detected as a container in first fragment and "team" is detected as a separate container from second fragment. But the system should also understand that both refer to the same container.

Coreference Resolution: Consider the problem text, "Jason had 49 quarters in his bank . His dad gave him 25 quarters . How many quarters does he have now ? ". Here, there is inherent ambiguity in the question. The “he” can refer to either Jason or His dad.

Container Detection: Consider the problem text, "Melanie picked 4 plums . Dan picked 9 plums and Sally picked 3 plums from the plum tree . How many plums were picked in total ?". Here, in the last fragment, "How many plums were picked in total ?" we detect no container which actually refers to "Melanie + Dan + Sally". Similarly, if “they” is detected as a container, we must detect who all does “they” refer to.

Verb Similarity: Consider the problem text, "Jason went to 11 football games this month . He went to 17 games last month and plans to go to 16 games next month . How many games will he attend in all ?". Here, the lemmatized verb in all the fragments except the question fragment w_x is "go" whereas the lemmatized verb in the question fragment w_x is "attend" and thus it is difficult to mark fragments which are associated with w_x.

Verb Categorization: If the category of the detected verb is incorrect, the quantities in the states are identified incorrectly which ultimately leads to an incorrect answer.

Conclusion

The problem of understanding and solving arithmetic word problems is not trivial. This is because it is not a single problem but has many subproblems like verb categorization, coreference resolution, dependency parsing etc. It is difficult to implement an algorithm which solves all different kinds of arithmetic word problems. We generalize the algorithm to cover as many models of word problems as possible. More rules can be added to extend the system to other word problems. For verb categorization we use the baseline approach i.e. using WordNet for categorization. Instead, training a machine learning model with the features extracted from the sentences to categorize verbs will improve accuracy.