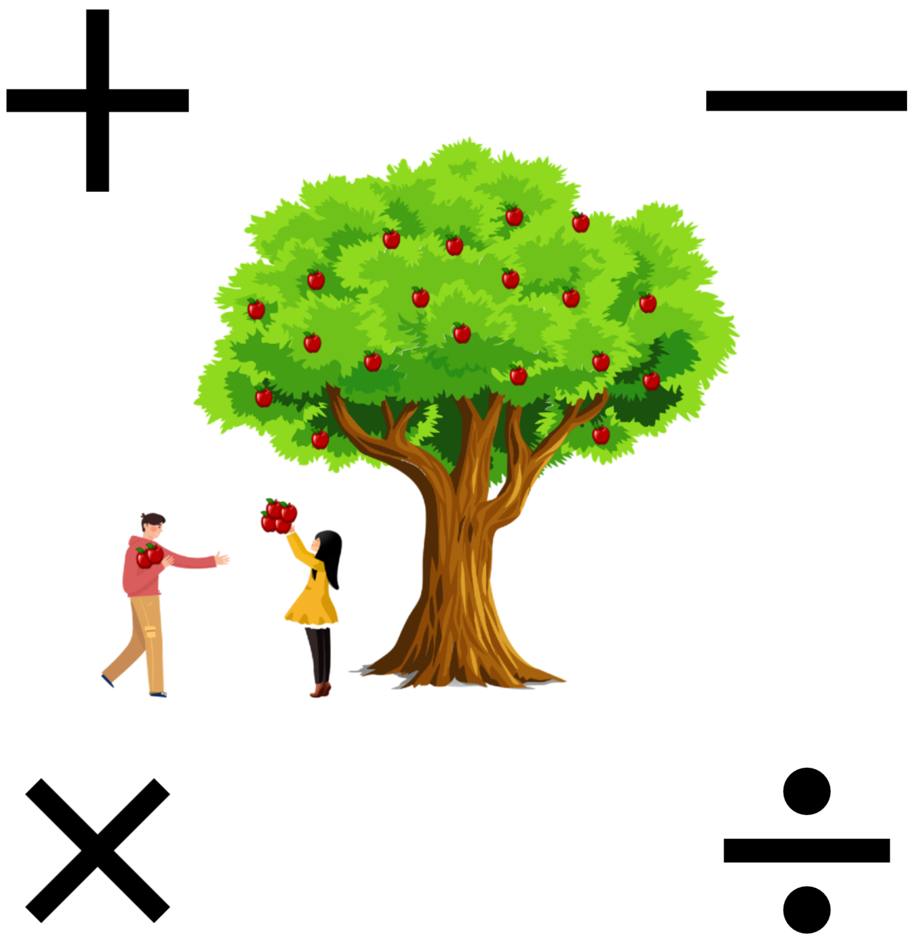
Solving math word problems with controlled language



Fabian A. Afatsawo

Layout: typeset by the author using LaTeX.
Cover illustration: by Fabian A. Afatsawo

Solving math word problems with controlled language

Translating Natural Language to Controlled Natural Language and performing calculations on word problems.

Fabian A. Afatsawo
12105155

Bachelor thesis
Credits: 18 EC

Bachelor *Kunstmatige Intelligentie*

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
Dr. Giovanni Sileno

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

Semester 1, 2021

Chapter 1

Abstract

To this day, simple middle school math word problems are difficult to solve for a computer. These problems are written in natural language and for this reason, are not easily parsable into first-order logic. However, controlled natural language is understandable by humans and easily parsable into first-order logic because of its rule-based nature. Can simple math word problems be solved by utilising a controlled natural language? What are the limits of such a system? What are the different types of problems it can solve? This paper tries to tackle this problem by implementing a controlled natural language as an intermediate step in calculating math problems written in natural language. Thereby, we decompose the problem in two steps: the use of machine learning-based machine translation will be explored to translate natural language to controlled natural language, and the use of a dedicated computational module for the calculation producing the output. Even if not reaching relevant performance on datasets used for state-of-the-art research with transformers, the proposed method produces satisfactory results on a limited subset of problems, showing the potential of being useful to reduce the gap.

Contents

Chapter 2

Introduction

Since the invention of the calculator, computers do an excellent job in calculating math expressions. The language for expressions is machine-understandable such that a system could perform calculations. However, typically the language used to describe math problems is a substrate of natural language (NL), which, although it relies on technical terms and grammatical conventions, is not machine-understandable. Here, we will focus on mathematical problems presented using natural language. How to make it computable? Well, we could consider a controlled natural language (CNL) which is a subset of NL that is human and machine-understandable. A controlled natural language for mathematics (CLM) is developed [15] to solve this problem. This CLM is used as a step towards automatic formalisation and verification of textbook mathematical text. However, the coverage of CLM facilitates common reasoning patterns but is limited. Furthermore, there is a gap. We consider math word problems written in NL and this CLM includes symbols. The task of converting NL to First-Order Logic (FOL) or FOL-like machinery remains an area of research. Neural models demonstrated difficulties in modelling NL to FOL [26]. Manual approaches are too costy. The use of CNL could assist to form a bridge in this task. Research supports this approach for instance [24], where simplified English texts are converted from NL to Logic using CNL as an intermediate step. Also, the conversion of legal text to CNL is attempted using Natural Language Processing techniques (NLP) [10].

For one-step approaches, attempts to solve school math problems have been made by using autoregressive language models like GPT-3 [8]. A significant amount of research is focussing on converting NL straight into logic. Unfortunately, the results are still unsatisfiable. To tackle this, targeting primarily a conversion of NL to CNL might improve performance in this task [14]. Because CNL has a direct logical interpretation, it allows solving simple math problems given in textual form by using logic. These simple problems consist of eg. additive and subtractive operations relative to objects and agents. Therefore, the core step of this method is the machine translation task from NL to CNL. The task requires a CNL that is suited for making calculations. Attempto Controlled English (ACE) is a possible CNL that can make this conversion [12]: ACE has a countable property and thus could be used to derive simple calculations.

Chapter 3

Background & Related work

3.1 Understanding and answering

Understanding human language and answering questions by this understanding is an overarching domain of research in AI being pursued by fields such as information retrieval (IR), automated reasoning and logic, and deep learning methods, e.g. with language models like GPT-3. Answering (querying) without in-depth understanding is the typical domain of IR, focusing on the quest of finding specific information in data through statistical and other numeric techniques. Answering through models is the domain of automated reasoning and logic; reasoning can be defined as the process of drawing conclusions with the help of existing knowledge, which in part need to be extracted from the given text. GPT-3 and the like seem to mimic human language as much as apparently performing some form of reasoning [7]. Some of the tasks GPT-3 is capable of doing are e.g. writing code, composing an email and having conversations [21]. Some form of understanding and answering are required for these tasks.

3.2 Controlled natural language

3.2.1 Definition and use cases

Controlled natural languages (CNLs) are subsets of natural languages. These CNLs are obtained by restricting grammar and vocabulary. Restrictions reduce the ambiguity and complexity of a language. CNLs have been used in the fields of software specification, theorem proving, text summaries, ontologies, rules, querying, medical documentation and planning.

1. Every man is human.
2. A woman is a human.
3. There is a man. He has 2 apples.
4. An athlete runs 60 minutes. If the athlete runs 10 minutes then the athlete runs 3 laps. How many laps does the athlete run?
5. A year has 365 days. Each day has 24 hours. Each hour has 60 minutes. Each minute has 60 seconds. How many seconds does a year have?

Figure 3.1: Examples of ACE expressions.

3.2.2 Attempto Controlled English

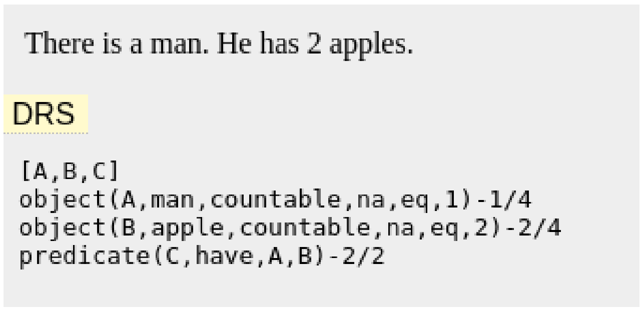
Attempto Controlled English (ACE) is a controlled natural language for the English language. A small set of construction and interpretation rules cause ACE to have a restricted syntax and semantics. Since it is a subset of natural language, ACE can be understood by any speaker of English. The language is developed by the University of Zurich since 1995 [3]. Besides the small rule set of ACE, it has enough depth to describe something as complex as ACE itself [17]. Whereby a full paper about ACE is written in ACE.

ACE features

ACE has a set of construction and interpretation rules. Which gives it a couple of features. Some of the associated requirements are: Every ACE text is written in the present simple tense. The use of other tenses makes it therefore not ACE compliant. Every noun refers to an object. ACE has a countable property which makes it able to concatenate objects. ACE text can be either one or multiple sentences, even including questions. Figure 3.1 shows examples of this in 4 and 5. In the case of multiple sentences, references can be made to parts of the text.

ACE Parsing Engine

The ACE Parsing Engine (APE) parses ACE compliant text. ACE is developed in rule-based fashion rather than machine learning based. APE is written in Prolog, as Prolog is a traditional choice for symbolic natural language processing applications. APE can translate ACE text into discourse representation structures (DRS). This DRS uses a variant of first-order logic. This translation allows to reason about the text.


There is a man. He has 2 apples.

DRS

```
[A,B,C]
object(A,man,countable,na,eq,1)-1/4
object(B,apple,countable,na,eq,2)-2/4
predicate(C,have,A,B)-2/2
```

Figure 3.2: APE output for the text "There is a man. He has 2 apples.", as per online APE client [20]

.

Figure 3.2 shows an example via the usage of APE. The text 'There is a man. He has 2 apples' is ACE compliant. Therefore, the APE can translate this text into a DRS format. In this example, there are 2 distinct nouns: the man and apples. Objects refer to nouns and their attributes. These objects come back in the DRS. We see that the APE uses the countable property to indicate that there are 2 apples. Actions and relations between objects are indicated by predicates. In this example, the predicate 'have' shows a relation between the objects.

3.3 Machine translation

The aim of machine translation is the task of converting a text from one language to another. Approaches taken include rule-based, statistical and machine learning-based methods. Rule-based machine translation uses linguistic information from dictionaries and grammars to produce a translation. Statistical machine translation uses statistical models whose parameters are derived from the analysis of bilingual text corpora. Machine learning-based, or neural machine translation, uses artificial neural networks to predict the likelihood of a sequence of words as means of translation. As previously stated, the task of translating NL into CNL has been done in rule-based fashion by [8] and [10]. This paper, therefore, will explore machine learning-based machine translation for this task.

3.3.1 Transformers

Transformers are a special type of neural network that adopts a mechanism of self-attention. This attention mechanism looks at an input sequence and decides which other parts of the sequence are important [2]. They are designed to handle sequential input data such as natural language. Transformers are used to solve complex language problems like Machine Translation, Question Answering, Chatbots, Text Summarization, etc. Transformers make use of Sequence-to-sequence learning (Seq2Seq) [16]. This Seq2Seq is about training models to convert sequences from one domain to another domain. The architecture of a Seq2Seq model often consists of an Encoder and a Decoder. The Encoder maps the input sequence into a high dimensional vector. The Decoder will turn this vector into the output sequence. Some transformers will be explained further in this paper.

3.3.2 Bidirectional transformers

BERT

Bidirectional Encoder Representations from Transformers (BERT) is a self-supervised pretraining technique [9]. It learns by intentionally masking sections of text and making predictions on what these sections should have been. When masking a section, the model uses both the left and right context of the masked section to learn. Because it uses both contexts, it is called bidirectional. When BERT was first released in 2018 it achieved state-of-the-art results on many NLP benchmark datasets.

RoBERTa

RoBERTa is built on BERT with a modified training procedure [28]. This includes removing BERT's next-sentence pretraining objective, and training in much larger batches. RoBERTa can be used to create sentence embeddings. These embeddings can be used for downstream tasks as finding semantic similarity.

3.3.3 Autoregressive transformers

GPT

Generative Pre-trained Transformer (GPT) models, developed by OpenAI, introduced powerful language models [1]. These models are so-called autoregressive. Such a model predicts future values based on past values. Therefore, an autoregressive model can be seen as a model that utilises previous predictions to generate new predictions.

OpenAI is developing a range of GPT models, where the models get increasingly bigger per generation. The models are trained on bigger datasets and internally increase in the number of parameters. The third-generation language prediction model in the GPT-n series is GPT-3, with the largest model containing 175B parameters. As previously stated in section 3.1, GPT-3 is able to perform a range of generative language tasks. In addition, GPT-3 can solve math word problems to some extent [8].

BART

Bart is an example of Seq2Seq [19]. The model architecture of BART consists of a transformer encoder-decoder (Seq2Seq) model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder. BART is pre-trained in two ways. First, by corrupting text with an arbitrary noising function. Second, by reconstructing the original text.

Chapter 4

Method

4.1 Architecture

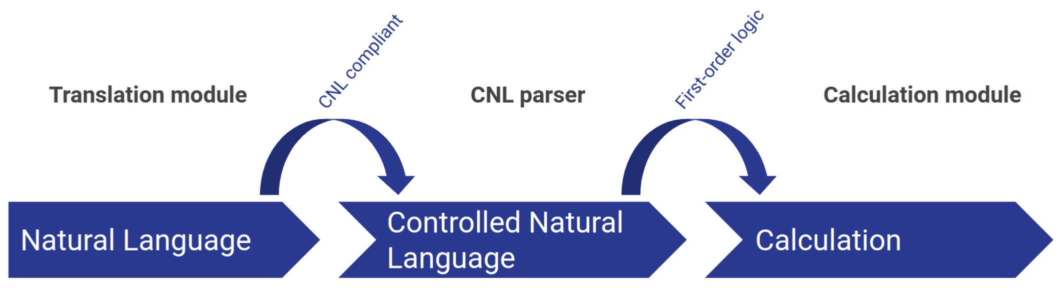


Figure 4.1: Proposed system architecture containing 3 modules: Translation module for converting NL to CNL, CNL parser to convert CNL to logic and Calculation module to derive solution from logic.

4.1.1 Machine Translation

The task of the machine translation module is to transform NL into CNL. A transformer encoder-decoder, sequence-to-sequence, model is able to perform this task. The training of such a model will be conducted in a supervised manner. Thus, requires a labelled dataset. This data consists of pairwise NL and their CNL counterparts.

4.1.2 CNL parser

Controlled natural language is machine-understandable, whereas natural language is not in most cases. The conversion made by the machine translation module needs to be checked for correctness. If the module fails to convert NL into a CNL form, then the problem can not be processed. A CNL parsing engine will ensure this check. If the parsing is successful, the parsing engine should return a logic-based representation of the problem.

4.1.3 Calculation module

A logic-based representation of the problem at hand will be returned by the CL parsing engine once the problem is CL compliant. This representation will serve as an input for the calculation module. Since the patterns to be found are expected to be rather fixed in a linguistic domain as math problems, certain rules can be established to perform calculations.

4.2 Data

4.2.1 Training Data

To train the machine translation module, NL and CNL couplings are needed. The target textual mathematical problems consist of different arithmetic operations. Addition, subtraction, division, multiplication as well as combinations of these. To form couplings, the CNL problems can be transformed into NL. This can be done by substitution of subjects or by paraphrasing the sentences. Furthermore, different tenses and the composition of sentences can broaden the dataset vertically as well as by horizontal composition.

Interestingly, since CNL compliant sentences are a subset of NL, the sentences themselves can be used in identity-identity training.

4.2.2 Target Application Data

Primary school exercises

The text on which we focus in this work are simple mathematical word problems [18]. Primary school math exercises form a perfect example for this. These problems contain e.g. adding to 1000, subtracting 3 numbers, multiplying 2 integers and dividing 3 integers.

GSM8K Dataset

The GSM8K dataset [13] consists of 8.5K high quality linguistically diverse grade school math word problems created by human problem writers [8]. The problems can be solved in 2 to 8 steps. The solutions involve performing a sequence of basic arithmetic operations. The dataset includes both questions and answers written in natural language. Figure 4.2 shows an example the dataset. Since the dataset is not CNL compliant it can not be used for training in machine translation. However, the dataset can be used in the evaluation of the system.

Question

> "Tim grows 5 trees. Each year he collects 6 lemons from each tree. How many lemons does he get in a decade?"

Answer

> "He gets 5*6=30 lemons per year.
> So he gets 30*10=**300** lemons in a decade."
> #### 300

Figure 4.2: Example problem from the GSM8K test set.

4.3 Evaluation

The system will be evaluated as a whole and on the individual parts.

4.3.1 Correct solution

To evaluate the whole system, the best metric is a correct solution. With a non-CNL compliant (or CNL compliant) problem as input and a correct solution as output, that defines the system as working as intended. The translation module transforms the problem into an CNL compliant problem. Then, the CNL parser converts this to a logistical representation. Finally, this representation can be understood and acted upon by the ad-hoc module, resulting in a solution.

4.3.2 CNL compliance

The machine translation module must be able to transform NL to CNL. If not, the problem can not be understood and solved by the system. The controlled language compliance shows what percentage of NL is being transformed to CNL. Therefore, this metric is important in evaluating the machine translation module. The higher the compliance, the more problems could be solved.

4.3.3 Semantic similarity

To further evaluate the translation module, one could look at the semantic similarity. Semantic similarity is a score based on how similar sentences are in terms of their meaning. In other words, after the translation module, the CNL compliant problem must be semantically similar to its input. This score could be generated in different ways. For example, by comparing sentence embeddings. These embeddings represent sentences and their semantic information as vectors. To approach this score, one could use the cosine similarity between sentence embeddings.

4.3.4 String edit distance

Besides meaning, one could look at the string edit distance. This distance is a way of quantifying how dissimilar two strings are. The minimum amount of operations required to transform one string into the other will define this distance. A large distance means that the string is changed greatly. In contrast, a small distance means that the strings are very similar.

Chapter 5

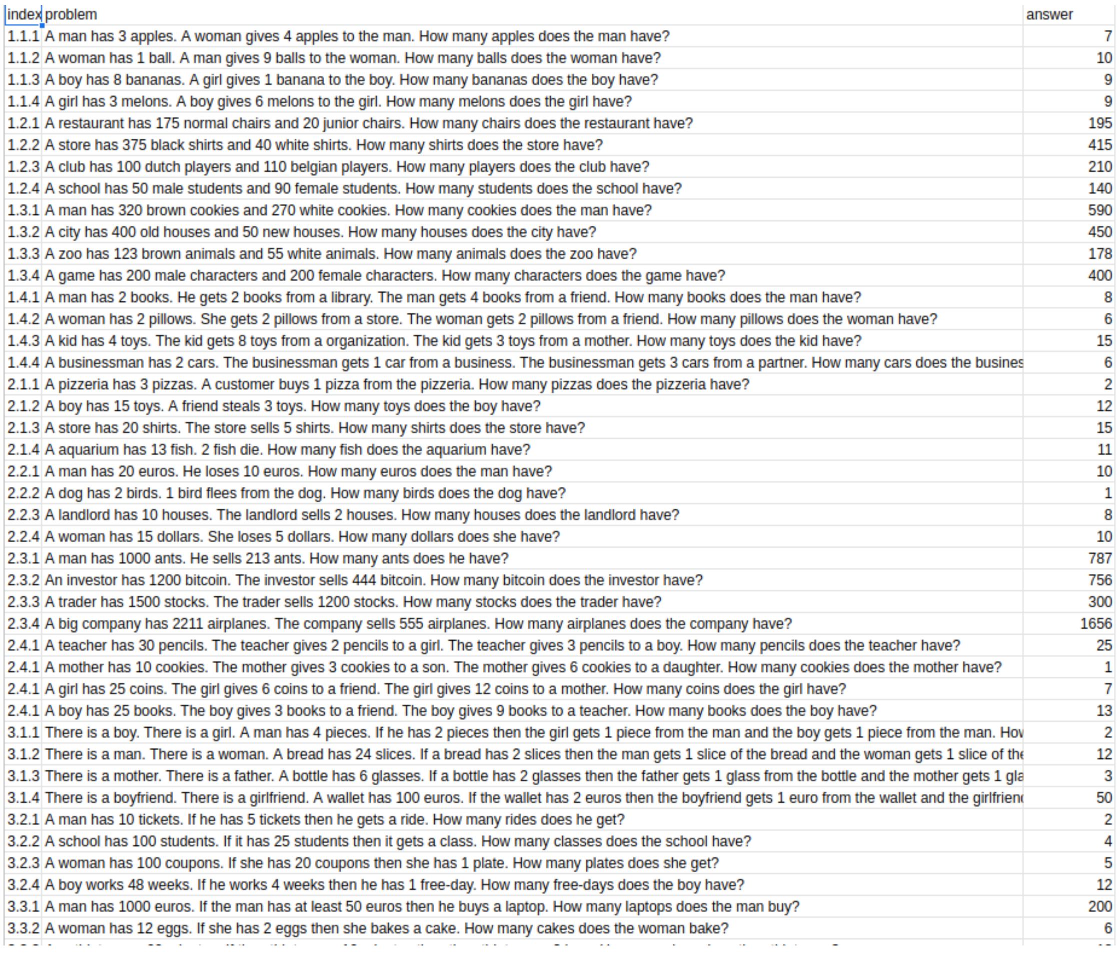
Implementation

5.1 Controlled natural language

The controlled language used in the implementation is Attempto Controlled English (ACE) [3]. Since ACE has a countable property capturing also cardinalities, it could be used to derive simple calculations.

5.2 Data structure

Math word problems written in ACE are not readily available. Therefore, a custom dataset has to be manually generated. A small dataset of 80 different ACE compliant math problems are created for the project. These 80 problems contain different mathematical operations. For example addition up to 100, subtraction of 3 numbers and multiplication of 2 digit integers [18]. In some of these problems, horizontal and vertical composition are present. Figure 5.1 shows an example of some of these problems. A paraphraser (Parrot [22]) is used to expand this dataset. Every sentence of a word problem gets paraphrased. Permutations of sentences and their paraphrased counterparts will form new problems. Furthermore, the 80 ACE problems are being translated into different tenses. These tenses contain present simple and past simple. The answers to these problems are known because they are self-generated. Therefore this custom dataset will not only be used to train the translation module but also to evaluate the whole system.



index	problem	answer
1.1.1	A man has 3 apples. A woman gives 4 apples to the man. How many apples does the man have?	7
1.1.2	A woman has 1 ball. A man gives 9 balls to the woman. How many balls does the woman have?	10
1.1.3	A boy has 8 bananas. A girl gives 1 banana to the boy. How many bananas does the boy have?	9
1.1.4	A girl has 3 melons. A boy gives 6 melons to the girl. How many melons does the girl have?	9
1.2.1	A restaurant has 175 normal chairs and 20 junior chairs. How many chairs does the restaurant have?	195
1.2.2	A store has 375 black shirts and 40 white shirts. How many shirts does the store have?	415
1.2.3	A club has 100 dutch players and 110 belgian players. How many players does the club have?	210
1.2.4	A school has 50 male students and 90 female students. How many students does the school have?	140
1.3.1	A man has 320 brown cookies and 270 white cookies. How many cookies does the man have?	590
1.3.2	A city has 400 old houses and 50 new houses. How many houses does the city have?	450
1.3.3	A zoo has 123 brown animals and 55 white animals. How many animals does the zoo have?	178
1.3.4	A game has 200 male characters and 200 female characters. How many characters does the game have?	400
1.4.1	A man has 2 books. He gets 2 books from a library. The man gets 4 books from a friend. How many books does the man have?	8
1.4.2	A woman has 2 pillows. She gets 2 pillows from a store. The woman gets 2 pillows from a friend. How many pillows does the woman have?	6
1.4.3	A kid has 4 toys. The kid gets 8 toys from a organization. The kid gets 3 toys from a mother. How many toys does the kid have?	15
1.4.4	A businessman has 2 cars. The businessman gets 1 car from a business. The businessman gets 3 cars from a partner. How many cars does the busines	6
2.1.1	A pizzeria has 3 pizzas. A customer buys 1 pizza from the pizzeria. How many pizzas does the pizzeria have?	2
2.1.2	A boy has 15 toys. A friend steals 3 toys. How many toys does the boy have?	12
2.1.3	A store has 20 shirts. The store sells 5 shirts. How many shirts does the store have?	15
2.1.4	A aquarium has 13 fish. 2 fish die. How many fish does the aquarium have?	11
2.2.1	A man has 20 euros. He loses 10 euros. How many euros does the man have?	10
2.2.2	A dog has 2 birds. 1 bird flees from the dog. How many birds does the dog have?	1
2.2.3	A landlord has 10 houses. The landlord sells 2 houses. How many houses does the landlord have?	8
2.2.4	A woman has 15 dollars. She loses 5 dollars. How many dollars does she have?	10
2.3.1	A man has 1000 ants. He sells 213 ants. How many ants does he have?	787
2.3.2	An investor has 1200 bitcoin. The investor sells 444 bitcoin. How many bitcoin does the investor have?	756
2.3.3	A trader has 1500 stocks. The trader sells 1200 stocks. How many stocks does the trader have?	300
2.3.4	A big company has 2211 airplanes. The company sells 555 airplanes. How many airplanes does the company have?	1656
2.4.1	A teacher has 30 pencils. The teacher gives 2 pencils to a girl. The teacher gives 3 pencils to a boy. How many pencils does the teacher have?	25
2.4.1	A mother has 10 cookies. The mother gives 3 cookies to a son. The mother gives 6 cookies to a daughter. How many cookies does the mother have?	1
2.4.1	A girl has 25 coins. The girl gives 6 coins to a friend. The girl gives 12 coins to a mother. How many coins does the girl have?	7
2.4.1	A boy has 25 books. The boy gives 3 books to a friend. The boy gives 9 books to a teacher. How many books does the boy have?	13
3.1.1	There is a boy. There is a girl. A man has 4 pieces. If he has 2 pieces then the girl gets 1 piece from the man and the boy gets 1 piece from the man. Ho	2
3.1.2	There is a man. There is a woman. A bread has 24 slices. If a bread has 2 slices then the man gets 1 slice of the bread and the woman gets 1 slice of th	12
3.1.3	There is a mother. There is a father. A bottle has 6 glasses. If a bottle has 2 glasses then the father gets 1 glass from the bottle and the mother gets 1 gla	3
3.1.4	There is a boyfriend. There is a girlfriend. A wallet has 100 euros. If the wallet has 2 euros then the boyfriend gets 1 euro from the wallet and the girlfrien	50
3.2.1	A man has 10 tickets. If he has 5 tickets then he gets a ride. How many rides does he get?	2
3.2.2	A school has 100 students. If it has 25 students then it gets a class. How many classes does the school have?	4
3.2.3	A woman has 100 coupons. If she has 20 coupons then she has 1 plate. How many plates does she get?	5
3.2.4	A boy works 48 weeks. If he works 4 weeks then he has 1 free-day. How many free-days does the boy have?	12
3.3.1	A man has 1000 euros. If the man has at least 50 euros then he buys a laptop. How many laptops does the man buy?	200
3.3.2	A woman has 12 eggs. If she has 2 eggs then she bakes a cake. How many cakes does the woman bake?	6
[illegible]	[illegible]	[illegible]

Figure 5.1: Examples of custom created word problems

5.3 Modules/Objects

5.3.1 Paraphraser

The paraphraser used is called Parrot [22]. Parrot is a paraphrase-based utterance augmentation framework purpose-built to accelerate training NLU models. Parrot contains a language model that is trained on multiple English paraphrases datasets [22]. A good paraphrase is adequate and fluent while being as different as possible on the surface lexical form. Parrot is able to change paraphrases in terms of three knobs: adequacy, fluency and diversity. Adequacy increases if the meaning is maintained. Fluency, according to an ancillary BERT model for Parrot [11], revers to the sentence being fluent in English. Diversity, according to Levenshtein distance [27] (section 5.3.5), revers to string difference between paraphrase and input.

In the implementation, the parameters of these knobs are dynamic. They are optimised by decreasing the order of intensity. The starting values are 90 per cent. The knobs are turned down sequentially by 5 per cent until the desired paraphrase is reached. The desired paraphrase is semantically equal but diverse with respect to the original sentence.

5.3.2 Translation module

The task of the machine translation module is to transform NL problems into ACE compliant problems. A sequence-to-sequence model could be used to perform this task. To train such a model, it needs pairwise data. In this case, it requires NL to ACE coupling data. A model that is able to do this is BART [19].

A Seq2Seq BART model, explained in section 3, is used as a translation module. A base model will be fine-tuned on the task of converting NL into ACE. The chosen base model is "facebook/bart-base" [5]. This BART model is base-sized and pre-trained in the English language. There is an even bigger model named BART large-sized [6], but such an investment of resources was not indicated for a proof-of-concept of the method. The training dataset is divided into train and test splits with an 80/20 ratio in a randomised fashion. The model is trained on 20 epochs.

The target of the translation is simple sentences because the ad-hoc module is based upon those DRS outputs. Therefore, already ACE compliant sentences can be transformed into simpler ACE compliant forms.

	Train	**Test**	**Past-to-Present**
Precision	0.70	0.63	0.71
Precision (lowercase)	0.77	0.73	0.77

Table 5.1: BART model precision on training dataset

Table 5.1 shows the precision of the trained BART model on different datasets.

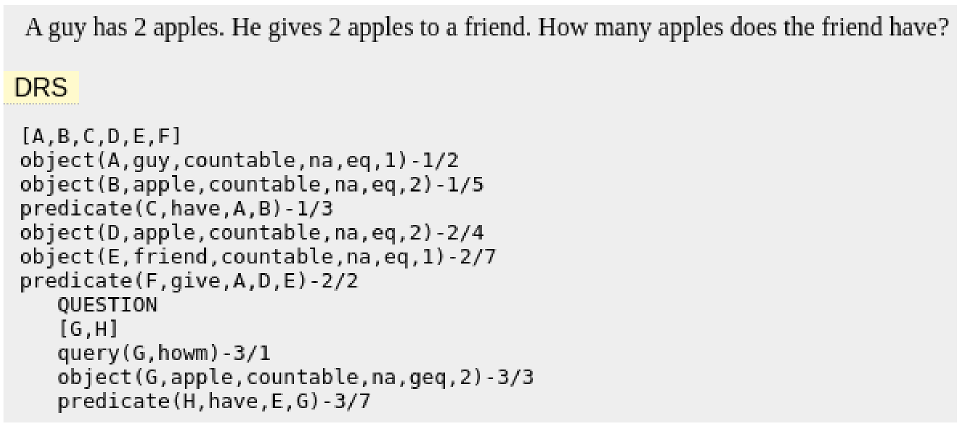
5.3.3 CNL parser

The ACE Parsing Engine (APE) is able to parse ACE compliant sentences. It will return an ACE compliant sentence in a discourse representation structure (DRS). If the input is not ACE compliant, it will fail to parse. This parser will be used in the system to check for ACE compliance and to transform it into DRS if possible.

An online client [20] is released as well as the code itself [4]. This code will be used to run APE locally.

5.3.4 Calculation module

The DRS output returned by the APE serves as an input for the calculation module. This DRS consists of a specific format. Therefore, a rule-based method is chosen for the calculation module. A method that can read this format and act upon it. Verbs have a unique meaning that leads to specific actions. These actions need to be implemented individually. A drawback of this is, that the module will lack an understanding of actions that are not implemented.



A guy has 2 apples. He gives 2 apples to a friend. How many apples does the friend have?

DRS

```
[A,B,C,D,E,F]
object(A,guy,countable,na,eq,1)-1/2
object(B,apple,countable,na,eq,2)-1/5
predicate(C,have,A,B)-1/3
object(D,apple,countable,na,eq,2)-2/4
object(E,friend,countable,na,eq,1)-2/7
predicate(F,give,A,D,E)-2/2
   QUESTION
   [G,H]
   query(G,howm)-3/1
   object(G,apple,countable,na,geq,2)-3/3
   predicate(H,have,E,G)-3/7
```

Figure 5.2: APE output for the text "A guy has 2 apples. He gives 2 apples to a friend. How many apples does the friend have?", as per online APE client [20]

.

Objects

The module extracts all objects and their respective attributes. Some of these attributes are id, name, amount and property. Agents are also objects, as shown in figure 5.2 where "guy" is an object. Every object has an inventory. This inventory can store other objects. In order words, each object can store any other object in their inventory.

Predicates

The actions that are performed are indicated by predicates. Predicates refer to objects and relationships. These predicates are implemented as rules in the calculation module. Figure 5.2 contains 3 predicates that will be explained in more detail.

First, "have" is implemented as store object B in agent A their inventory. So, the guy (agent A) will have 2 apples in his inventory.

Second, "give" will make sure that agent A gives object D to agent E. Here, object D will be taken out of the inventory of agent A and put into the inventory of agent E. So, the guy will now have 0 apples and the friend (agent E) has 2.

Finally, the query predicate "have" will request the inventory of agent E about object G. Thus, the amount of apples that the friend has in his inventory is being requested. Which equals to 2 apples.

5.3.5 Evaluation modules

Semantic similarity

A RoBERTa [28] model, explained in section 3, is used to calculate semantic similarity via cosine distance. The model used is 'stsb-roberta-large' [23]. This model is trained on a large English corpus.

String edit distance

The Levenshtein distance is used to calculate the string edit distance. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other. It is named after the Soviet mathematician Vladimir Levenshtein, who considered this distance in 1965 [27].

5.4 Process pipeline

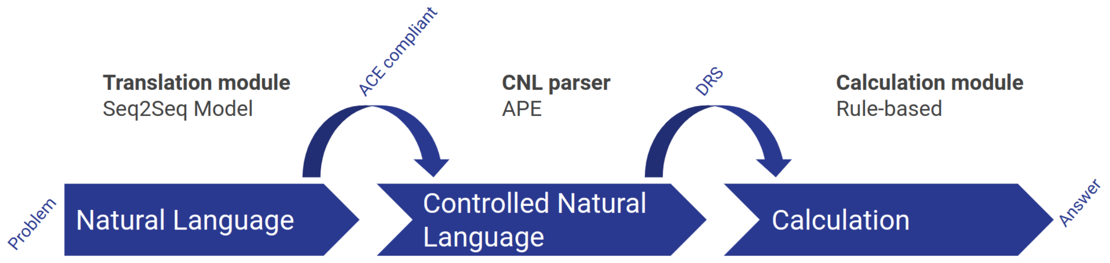
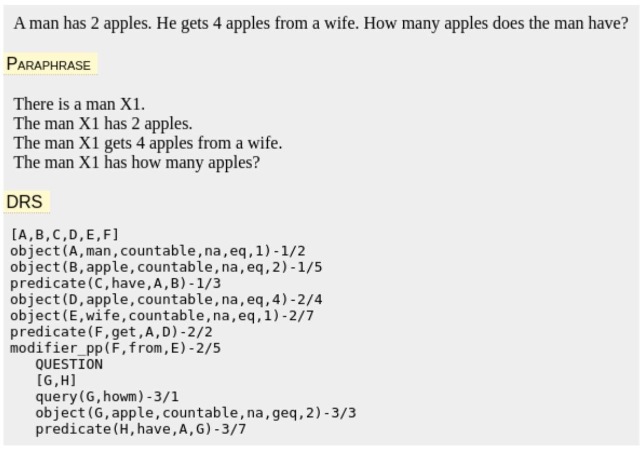


Figure 5.3: Architecture of implementation.

The complete process pipeline is shown in figure 5.3. First, the problem will go into the translation module. This BART model trained to convert NL into ACE will try to convert the problem into an ACE compliant problem. Next, the output of the translation module, the attempted ACE compliant text, will go into the APE parser. Once the ACE compliant problem is parsed by the APE then the DRS form of the problem will be returned. This DRS problem will be the input of the calculation module. The calculation module is an ad-hoc module written in Python that works in a rule-based fashion. Finally, the calculation module will return an answer.

5.5 Examples of execution

Figure 5.5 show an example of the system calculating a math problem correctly. The underlying DRS output generated by APE is shown in figure 5.4.



A man has 2 apples. He gets 4 apples from a wife. How many apples does the man have?

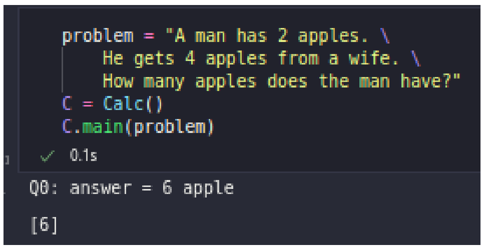
PARAPHRASE

There is a man X1.
The man X1 has 2 apples.
The man X1 gets 4 apples from a wife.
The man X1 has how many apples?

DRS

```
[A,B,C,D,E,F]
object(A,man,countable,na,eq,1)-1/2
object(B,apple,countable,na,eq,2)-1/5
predicate(C,have,A,B)-1/3
object(D,apple,countable,na,eq,4)-2/4
object(E,wife,countable,na,eq,1)-2/7
predicate(F,get,A,D)-2/2
modifier_pp(F,from,E)-2/5
   QUESTION
   [G,H]
   query(G,howm)-3/1
   object(G,apple,countable,na,geq,2)-3/3
   predicate(H,have,A,G)-3/7
```

Figure 5.4: APE output for the text “A man has 2 apples. He gets 4 apples from a wife. How many apples does the man have?”, as per online APE client [20].



```
problem = "A man has 2 apples. \
    He gets 4 apples from a wife. \
    How many apples does the man have?"
C = Calc()
C.main(problem)
```

✓ 0.1s

```
Q0: answer = 6 apple

[6]
```

Figure 5.5: Execution example of the system solving a math word problem.

Chapter 6

Results

6.1 Correct solution

Both the present simple and past simple datasets have similar scores shown in table 6.1. They score around 60 per cent on correct calculation. Looking at the ACE compliant translations they score around 70 per cent. The correct problems mostly contain simple addition, subtraction and multiplication problems. In opposition, division and combinatorial problems seem to be more difficult. Unfortunately, the system has a score of 0 per cent on the GSM8K test set as seen in table 6.2. Other evaluation steps will look deeper into this.

The 90 per cent of faulty ACE predictions on the GSM8K dataset are categorised in figure 6.1. The predictions are categorised by the errors the APE gives. We see for example that there are predictions where a sentence contains 'if' but not 'then'. Therefore, the APE will not approve such a sentence as ACE compliant.

6.2 CNL compliance

The CNL compliance, ACE compliance in this experiment, on the custom dataset is around 80 per cent shown in figure 6.2. This is quite high compared to the GSM8K dataset [13] where only 10 per cent is translated into ACE compliant text.

Group	**ACE compliant**	**Total**
Present simple - ACE	72%	61%
Past simple - Not ACE	71%	62%

Table 6.1: Custom dataset - Primary school exercises

Group	Score
9-12-year-olds	60%
OpenAI	55%
GPT-3	30%
System	0%

Table 6.2: GSM8K Dataset Test set scores

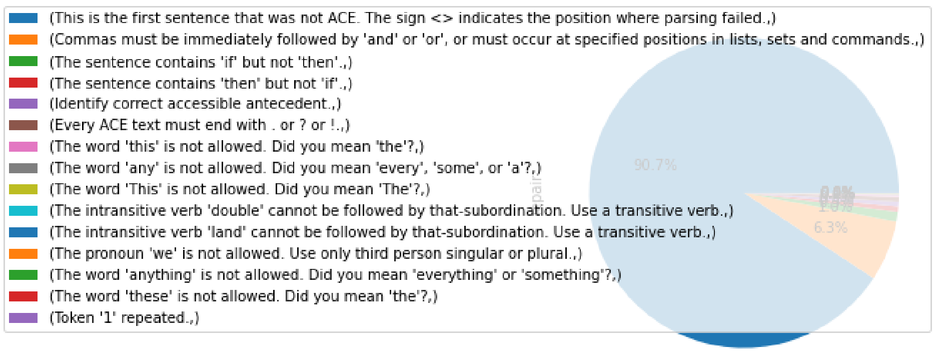


Figure 6.1: Overview on faulty ACE predictions on GSM8K dataset. The errors are generated by APE.

6.3 Semantic similarity

Figure 6.3 shows that the present simple and past simple datasets are highly semantically similar. On the other hand, translations on the GSM8K dataset are less semantically similar to their inputs.

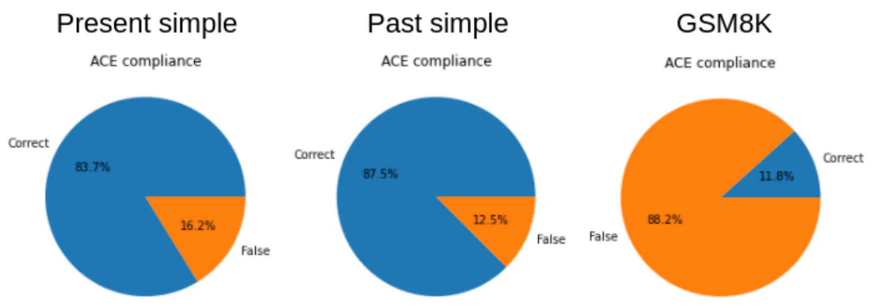


Figure 6.2: ACE compliance via APE.

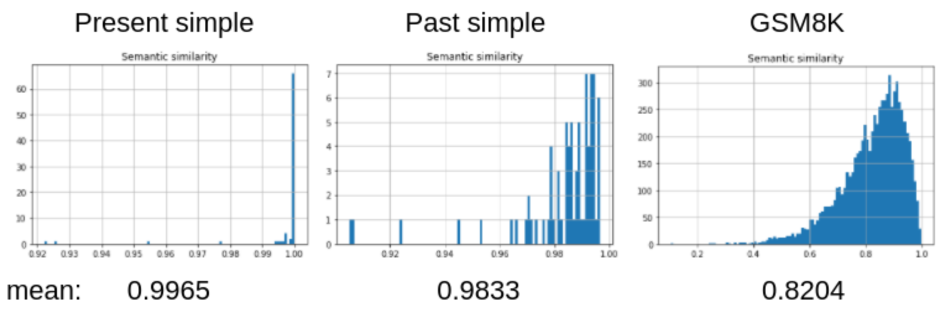


Figure 6.3: Semantic similarity via RoBERTa.

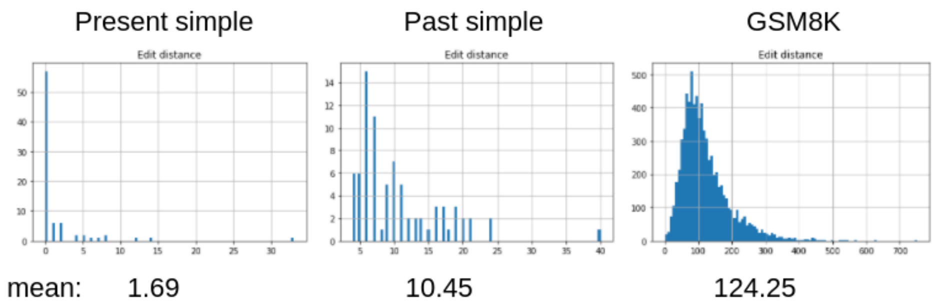


Figure 6.4: String edit distance via Levenshtein distance.

6.4 String edit distance

The present simple and past simple datasets have a low string edit distance as shown in figure 6.4. The amount of operations needed to transform the translated datasets into their original input is therefore low. On the other hand, translations on the GSM8K dataset have a high string edit distance. This means that the translations changed the original inputs in such a way that a lot of operations are needed to revert the process.

Chapter 7

Discussion

7.1 Translation module

To some extent, the translation module works as intended. Table 5.1 shows that the model is able to convert around 70 per cent of text in past tense (not ACE) to present tense (ACE). Table 5.1 also shows that 70 per cent of training and test set, of the custom dataset, gets translated to ACE compliant sentences.

Concerning the observed issues, the main cause is plausibly insufficient data. Especially the GSM8K dataset contains problems with different tenses, the problems are bigger in length and more complex. Therefore, by manually generating more ACE compliant problems as complex as in the GSM8K dataset this can be overcome.

7.2 Calculation module

7.2.1 Anaphoric referencing

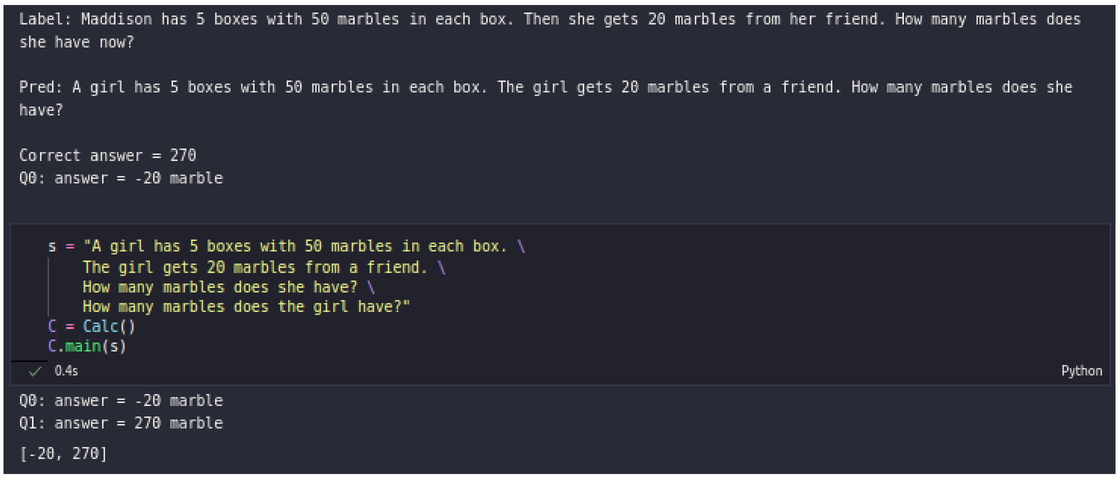
The main problem in the calculation module is referencing. The system is not able to link objects together when they are being referred to in a different way. For instance, when synonyms are used the calculation module refers to different objects, therefore the proposed solution will fail. Consider for instance the following problem:

> "There are 5 tables. Each table has 4 seats. How many chairs do the tables need?"

In this example, seats and chairs refer to the same object. The system does not know this and therefore will treat them as two different objects. Abstractions such as superclasses and collections also form a problem for the calculation module. The system does not have any understanding of taxonomy.

> "A neighbourhood has 12 rabbits. The neighbourhood has twelve cats. The city has 50 dogs. If there are 2 cats then the city gets 60 cats. How many pets does the neighbourhood have?"

In this example, the rabbits, cats, dogs, etc. all refer to pets. The system does not know this. The question at the end asks about the total number of pets. The system will fail because it does not have any knowledge about pets.



```
Label: Maddison has 5 boxes with 50 marbles in each box. Then she gets 20 marbles from her friend. How many marbles does
she have now?

Pred: A girl has 5 boxes with 50 marbles in each box. The girl gets 20 marbles from a friend. How many marbles does she
have?

Correct answer = 270
Q0: answer = -20 marble
```

```
s = "A girl has 5 boxes with 50 marbles in each box. \
    The girl gets 20 marbles from a friend. \
    How many marbles does she have? \
    How many marbles does the girl have?"
C = Calc()
C.main(s)
```

```
Q0: answer = -20 marble
Q1: answer = 270 marble
[-20, 270]
```

Figure 7.1: Anaphoric referencing limitation of APE.

Figure 7.1 shows a limitation of the APE. In this case, there are two agents a girl and a friend. In the question 'she' refers intentionally to the girl, however, the system refers to a friend. The system does this because APE refers to the last noun being used. In this case, the last noun being used is a friend. The system, therefore, gives a wrong answer. The example shows that when the girl explicitly is being referred to, the system will give a correct answer.

Knowledge base

A possible solution to the referencing problem could be integrating a knowledge base. Such a knowledge base would have a basic understanding of taxonomy. Therefore the use of synonyms and abstractions should not form a problem in the system.

7.2.2 Complexity

The GSM8K dataset [13] contains more complex problems than the training set. At the moment, the system does not have an understanding of fractions and percentages. On the other hand, the GSM8K dataset does contain these types of problems. By adding more support rules to the calculation module, this could be solved.

Chapter 8

Conclusion & Future works

The system can solve to a good extent simple mathematical word problems. Also because the system is trained on problems generated in that way. Problems containing addition, subtraction and multiplication are solvable when objects are being referred to directly. Division and combinatorial problems are harder to solve at the moment. The translation module is trained on little data, an expansion of sufficient data could improve the model. The problems in the calculation module are due to incorrect referencing and a lack of rules with respect to complexity.

8.1 Future works

8.1.1 Translation module

Expand training data

No training data made for this purpose has been found in the literature. Therefore, the training data used in this experiment is manually generated. This resulted in a small dataset. The expansion of training data could lead to a better evaluation of the system. This expansion consists of more patterns of sentences and other ways of combining them.

Other machine translation models

The language model used for the translation module in this experiment is BART [5]. Further research could explore different models. With respect to BART, there is a larger BART model [6] that could show improvements. This model could not be investigated due to a lack of resources. Other than BART, more sequence-to-sequence models could be explored [25].

8.1.2 Calculation module

Expand calculation module

The calculation module fails in complexity. Problems that require operations that the module does not possess are complex. By adding support for these operations, the module should be able to make these types of complex calculations.

Language model vs Rule-based

A rule-based module has its flaws. Rules need to be implemented, if not, the system will not have any knowledge on what to do. A language model dedicated to the logistical representation of a controlled language could resolve this problem. This model should have a knowledge base of a language. Therefore, this model will contain a taxonomy. Such a model should be trained on the logistical representation of a problem in controlled language and the respective solution to this problem.

8.1.3 Potential applications

CNL is the simplest language one could think of. The use of CNL could lead to clearer definitions of word math problems. These can be useful for children with autism who might be happy with clearly stated problems. Also, one could think of a reverse application where NL on which you write is a reflection of what the target person is able to read (i.e. style-based generation).

Bibliography

[1] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. URL: https://s3-us-west-2.amazonaws.com/openaiassets/research-covers/languageunsupervised/language understanding paper. pdf, 2018.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. arXiv preprint arXiv:1706.03762, 2017.

[3] Attempto Project Website. URL: http://attempto.ifi.uzh.ch/site/.

[4] Attempto/APE Github. URL: https://github.com/Attempto/APE.

[5] BART base English model. URL: https://huggingface.co/facebook/bart-base

[6] BART large English model. URL: https://huggingface.co/facebook/bart-large

[7] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

[8] Cobbe, K., Kosaraju, V., Bavarian, M., Hilton, J., Nakano, R., Hesse, C., & Schulman, J. (2021). Training Verifiers to Solve Math Word Problems. arXiv preprint arXiv:2110.14168.

[9] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[10] Dragoni, M., Villata, S., Rizzi, W., & Governatori, G. (2016). Combining NLP Approaches for Rule Extraction from Legal Documents. In Proceedings of the workshop on “Mining and Reasoning with Legal Texts” collocated at the 29th International Conference on Legal Knowledge and Information Systems.

[11] Fluency BERT model Parrot for English language. URL: https://huggingface.co/prithivida/parrot_fluency_on_BERT

[12] Fuchs, N. E., & Schwitter, R. (1996). Attempto controlled english (ace). arXiv preprint cmp-lg/9603003.

[13] GSM8K dataset. URL: https://github.com/openai/gradeschool-math

[14] Hales, T. (2019). An Argument for Controlled Natural Languages in Mathematics.

[15] Humayoun, M., & Rafalli, C. (2008). MathNat-Mathematical Text in a Controlled Natural Language. Special issue: Natural Language Processing and its Applications. Journal on Research in Computing Science, 46.

[16] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In Proceedings of NIPS, pages 3104–3112, 2014. URL: papers.nips.cc/paper/5346-sequence-tosequence-learning-with-neural.pdf.

[17] Norbert E. Fuchs, Kaarel Kaljurand, and Tobias Kuhn. ACE can be described by itself. In Simon Clematide, Manfred Klenner, and Martin Volk, editors, Searching Answers — Festschrift in Honour of Michael Hess on the Occasion of His 60th Birthday, pages 45–48. Monsenstein und Vannerdat, 2009.

[18] Marcus Guido (2021). 120 Math Word Problems To Challenge Students Grades 1 to 8. URL: https://www.prodigygame.com/main-en/blog/math-word-problems/

[19] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461, 2019.

[20] Online APE Client. URL: http://attempto.ifi.uzh.ch/ape/.

[21] Towards data science Blogpost (2020). The Mighty GPT-3. An overview of the capabilities of GPT-3. URL: https://towardsdatascience.com/the-mighty-gpt-3-1cb6ee477932

[22] Prithiviraj Damodaran (2021). Parrot: Paraphrase generation for NLU. URL: https://github.com/PrithivirajDamodaran/Parrot_Paraphraser.

[23] RoBERTa model stsb-roberta-large URL: https://huggingface.co/sentence-transformers/stsb-roberta-large

[24] Safwat, H., Zarrouk, M., & Davis, B. (2018). Rewriting simplified text into a controlled natural language.

[25] Sequence-to-sequence models created by simpletransofmers.ai. URL: https://simpletransformers.ai/docs/seq2seq-model/

[26] Singh, H., Aggrawal, M., & Krishnamurthy, B. (2020). Exploring neural models for parsing natural language into First-Order Logic. arXiv preprint arXiv:2002.06544.

[27] В. И. Левенштейн (1965). Двоичные коды с исправлением выпадений, вставок и замещений символов [Binary codes capable of correcting deletions, insertions, and reversals]. Доклады Академии Наук СССР (in Russian). 163 (4): 845–848. Appeared in English as: Levenshtein, Vladimir I. (February 1966). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady. 10 (8): 707–710. Bibcode:1966SPhD...10..707L.

[28] Yinhan Liu , Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. arxiv preprint arXiv:1907.11692.