Analytical Reasoning of Text

Anonymous ACL submission

Abstract

Analytical reasoning is an essential and challenging task that requires a system to analyze a scenario involving a set of particular circumstances and perform reasoning over it to make conclusions. However, current neural models with implicit reasoning ability struggle to solve this task. In this paper, we study the challenge of analytical reasoning of text and collect a new dataset consisting of questions from the Law School Admission Test from 1991 to 2016. We analyze what knowledge understanding and reasoning abilities are required to do well on this task, and present an approach dubbed ARM. It extracts knowledge such as participants and facts from the context. Such knowledge are applied to an inference engine to deduce legitimate solutions for drawing conclusions. In our experiments, we find that ubiquitous pre-trained models struggle to deal with this task as their performance is close to random guess. Results show that ARM outperforms pre-trained models significantly. Moreover, we demonstrate that ARM has better explicit interpretable reasoning ability. [1]

1 Introduction

Transformer-based pre-trained language models including BERT (Devlin et al., 2018), GPT-2 (Radford et al., 2019) and RoBERTa (Liu et al., 2019) have achieved state-of-the-art performance on a variety of NLP tasks. However, they still struggle to perform deep reasoning beyond shallow-level semantic understanding of literal clues. For example, Talmor et al. (2020) show that pre-trained models fail completely on half of eight reasoning tasks that require symbolic operations. We hope to challenge current systems and take a step further towards analytical reasoning.

Analytical reasoning assesses the ability of systems to *understand the knowledge, including participants, facts and literal rules mentioned in the context, perform reasoning over the extracted knowledge, and make conclusions.* In this paper, we study the challenge of analytical reasoning (AR). We collect a new dataset **AR-LSAT** from the Law School Admission Test[2] (LSAT) from 1991 to 2016 to facilitate research on analytical reasoning. An example of analytical reasoning in LSAT is given in Figure 1, whose task is to separate participants (i.e., *A,B, etc.*) into two positions (i.e., *X committee and Y committee*) under certain constraints. We can see that solving the problem requires a system to understand the knowledge in the context including participants, positions, rules expressed in natural language (e.g., "*If G serves on X, so does B*") and facts (e.g., "*D and F both serve on the X committee*"). Then, it needs to deduct logical expressions

[Grouping Game] Passage:
Seven directors -A, B, C, D, E, F, and G- serves on the X committee or the Y committee.

If A serves on X, then B serves on Y. **R-1**
If C serves on X, then D and E serve on Y. **R-2**
F serves on a different committee with G. **R-3**
E serves on a different committee with A. **R-4**
If G serves on X, so does B. **R-5** **Rules**

Question:
If D and F both serve on the X committee, **Fact**
then which one of the following could be true?

Options:
A. A and C both serve on the X committee.
   $(C\ on\ X)\&(D\ on\ X)$ conflict with R-2
B. A and E both serve on the Y committee.
   $(A\ on\ Y)\&(E\ on\ Y)$ conflict with R-4
C. B and G both serve on the X committee.
   $(G\ on\ X)\&(F\ on\ X)$ conflict with R-3
D. C and E both serve on the Y committee. √
E. G and E both serve on the X committee.
   $(G\ on\ X)\&(F\ on\ X)$ conflict with R-3

Participants	**Positions**	**Fact**
(A,B,C,D,E,F,G)	(X,Y)	$(D\ on\ X)\&(F\ on\ X)$

Rules to Logical Expressions
R-1: $A\ on\ X \rightarrow B\ on\ Y$
R-2: $C\ on\ X \rightarrow (D\ on\ Y)\&(E\ on\ Y)$
R-3: $Position\ of\ F \neq Position\ of\ G$
R-4: $Position\ of\ E \neq Position\ of\ A$
R-5: $G\ on\ X \rightarrow B\ on\ X$

Figure 1: An example of the required reasoning process to do well on the AR task. The input is a passage, a question and multiple options, and the output is the most plausible answer.

[1] All our data and codes will be released upon acceptance.

[2] https://en.wikipedia.org/wiki/Law_School_Admission_Test

(e.g., “*G on X* $\rightarrow$ *B on X*") from the rules, and draw inference before making conclusions.

In this paper, we analyze the knowledge understanding and reasoning ability required for solving this task and present Analytical Reasoning Machine (ARM), a framework that can comprehend the context and perform reasoning for making a conclusion. It extracts participants, rules and facts described in the context of text. Each literal rule is mapped into an executable logical constraint function, which assesses whether a solution satisfies a particular rule. With such logical-level understanding, ARM is capable of deducing a group of legitimate solutions for the question and select the most plausible option as the answer.

Experiments show that pre-trained models struggle to learn this task, which indicates that this task is very challenging for current models as it requires the complex reasoning ability far beyond implicit reasoning over the literal clues. Our system outperforms pre-trained models significantly. Further analysis demonstrates that our system has better interpretability. The contributions are threefold.

- We collect a new dataset AR-LSAT to facilitate research on analytical reasoning.
- We present a reasoning framework that can comprehend the context and perform explicit interpretable reasoning to draw conclusion.
- Experiments indicate that this task is challenging and our system outperforms pre-trained models significantly.

2 Related Works

There is an increasing trend on machine reasoning research in recent years. The reasoning ability investigated are partitioned into several major aspects, including (1) logical reasoning; (2) commonsense reasoning; (3) mathematical reasoning and (4) multi-hop reasoning.

Logical Reasoning The task of Natural Language Inference (NLI) (Dagan et al., 2005; Bowman et al., 2015; Wang et al., 2018; Williams et al., 2018; Welleck et al., 2018; Khot et al., 2018; Nie et al., 2019; Bhagavatula et al., 2019; Liu et al., 2020a) requires the models to detect the logical entailment relationship of two sentences. There have been Machine Reading Comprehension (MRC) datasets (Rajpurkar et al., 2016; Welbl et al., 2017; Yang et al., 2018a; Huang et al., 2019b) that examine the ability of logical reasoning. LogiQA (Liu et al., 2020b) and ReClor (Yu et al., 2020) are sourced from examination in realistic scenario and examine a range of logical reasoning skills.

Commonsense Reasoning There are many recent benchmarks that assess the commonsense reasoning capabilities from different aspects, like social (Rashkin et al., 2018), physics (Talmor et al., 2018; Zellers et al., 2019), or temporal (Zhou et al., 2019) aspects. There exist several MRC datasets that require commonsense knowledge (Ostermann et al., 2018; Zhang et al., 2018; Huang et al., 2019a).

Mathematical Reasoning There are many existing datasets (Kushman et al., 2014; Hosseini et al., 2014; Koncel-Kedziorski et al., 2015; Clark et al., 2016; Ling et al., 2017) that focus on mathematical word problems. Ling et al. (2017) builds a dataset that encourages generating answer rationales beyond simply selecting the correct answer. DROP (Dua et al., 2019) is a benchmark MRC dataset requiring mathematical reasoning. Saxton et al. (2019) focuses on algebraic generalization.

Multi-hop Reasoning Multi-hop reasoning over textual data (Talmor and Berant, 2018; Welbl et al., 2018; Yang et al., 2018b; Inoue et al., 2020) requires a model to reason over multiple paragraphs before making prediction.

To the best of our knowledge, there has not an existing benchmark dataset that completely focuses on the analytical reasoning over textual data. We introduce a new dataset to fill this gap and to foster research on this area.

3 Task and Dataset

In this section, we describe the task of analytical reasoning and introduce the dataset AR-LSAT we collected from the Law School Admission Test.

3.1 Task: Analytical Reasoning of Text

Taking a passage, a question, and multiple options as the input, a system is required to select the most plausible answer as the output. Each passage describes a reasoning game belonging to various types, including three dominant types: **ordering games**, **grouping games**, and **assignment games**, which are described as follows and examples are given in Figures 1 and 2:

- **Ordering games** are to order participants based on given facts and rules.

[Ordering Game] Passage
A professor must determine the order in which five of her students - Fernando, Ginny, Hakim, Juanita, and Kevin- will perform in a recital.
Ginny perform earlier than Fernando. **R-1**
Kevin perform earlier than Hakim and Juanita. **R-2**
Hakim perform either immediately before or immediately after Fernando. **R-3**
Question
Which one of the following could be the order the students perform?

Options
A. Ginny, Fernando, Hakim, Kevin, Juanita ×R-2
B. Ginny, Juanita, Kevin, Hakim, Fernando ×R-2
C. Ginny, Kevin, Hakim, Juanita, Fernando ×R-3
D. Kevin, Ginny, Juanita, Fernando, Hakim√
E. Kevin, Juanita, Fernando, Hakim, Ginny ×R-1

Fact
Uncertain

Positions
$(1^{st}, 2^{nd}, 3^{rd}, 4^{th}, 5^{th})$

Participants
$(Fernando, Ginny, Hakim, Juanita, Kevin)$

Rules to Logical Expressions
R-1: $Pos.of\ Ginny < Pos.of\ Fernando$
R-2: $(Pos.of\ Kevin < Pos.of\ Hakim)$ & $(Pos.of\ Kevin < Pos.of\ Juanita)$
R-3: $(Pos.of\ Hakim = Pos.of\ Fernando + 1) | (Pos.of\ Hakim = Pos.of\ Fernando - 1)$

[Assignment Game] Passage
Five cashiers-Adams, Bates, Cox, Drake, and Edwards-each of whom works alone on exactly one day, Monday through Friday
Adams will work only on Tuesday or Thursday. **R-1**
Bates will not work on Monday or Wednesday. **R-2**
Cox works on Friday. **F-1**
Edwards don't work next to Drake **R-3**
Question
Which one of the following is a possible work schedule?

Options
A. Edwards, Bates, Adams, Drake, Cox ×R-1
B. Drake, Adams, Bates, Edwards, Cox ×R-2
C. Edwards, Adams, Cox, Bates, Drake ×F-1
D. Edwards, Adams, Drake, Bates, Cox √
E. Drake, Edwards, Bates, Adams, Cox ×R-3

Fact
Cox on Fri.

Participants
$(Adams, Bates, Cox, Drake, Edwards)$

Positions
$(Mon., Tues., Wed., Thur., Fri.)$

Rules to Logical Expressions
R-1: $Adams\ on\ Tues. | Adams\ on\ Thur.$
R-2: $\neg(Bates\ on\ Mon. | Bates\ on\ Wed.)$
R-3: $Pos.of\ Edwards \neq Pos.of\ Drake + 1$

Figure 2: Examples of ordering game and assignment game in AR task. Facts and Rules are highlighted in orange and blue, respectively. Example of grouping game is shown in Figure 1. × indicates conflict.

- **Grouping games** are to separate participants into groups with given facts and rules.
- **Assignment games** are to assign characteristics to the participants with given rules, like assigning schedules for people.

3.2 Dataset: AR-LSAT

We collect data from nearly 90 LSAT exams from 1991 to 2016 and select questions from the analytical reasoning part to construct the dataset, dubbed **AR-LSAT**. Each exam in LSAT consists of 101 questions, 24 of which are AR questions. We finally leave up the questions with 5 answer options. The statistics are shown in Table 1. We manually categorize and analyze question types in AR-LSAT according to different reasoning types, and describe the detailed descriptions and corresponding examples in the Appendix D.

Number of questions	2,046
Average length of passages	99.3
Average length of questions	19.1
Average length of answers	6
Number of options	5
Ratio of ordering game	42.5%
Ratio of grouping game	38.75%
Ratio of assignment game	18.75%

Table 1: Data statistics of AR-LSAT dataset.

3.3 Baseline: Pre-trained Model

Pre-trained Transformer (Vaswani et al., 2017) based language models achieved impressive performance on a wide variety of tasks. There are several representative pre-trained models, like BERT (Devlin et al., 2018), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2019). We employ these powerful pre-trained models as our baselines after being fine-tuned on our dataset. Specifically, we take the concatenated sequence $X = \{[CLS], passage, [SEP], question, option\}$ as the input, where $[CLS]$ is the ending special token and $[SEP]$ is used to split two types of input. The final hidden vector at [CLS] is taken for classification. However, we find that these models struggle to deal with this task as their performances are close to random guess. For example, RoBERTa achieves 23.1% accuracy on the test set.

3.4 Challenges

In this part, we point out the reasoning ability required for solving AR questions, and put forward the challenges that systems should face.

As we can observe from the examples in Figure 1 and Figure 2, AR questions test a range of reasoning skills:

1) Comprehending the knowledge including participants of events, facts, and rules described in the context.

2) Extracting machine-understandable logical functions (expressions) from the rules. For example, the rule "*If A serves on X, then B serves on Y.*" needs to be transferred as logical expression "$A\ on\ X \rightarrow B\ on\ Y$",

3) Making deductions to derive legitimate solutions that satisfy extracted logical functions.

4) Selecting the answer that satisfies all the rules with the deducted legitimate solutions. In the examples, a system should eliminate options that conflict with rules and select the option that accords with legitimate solutions.

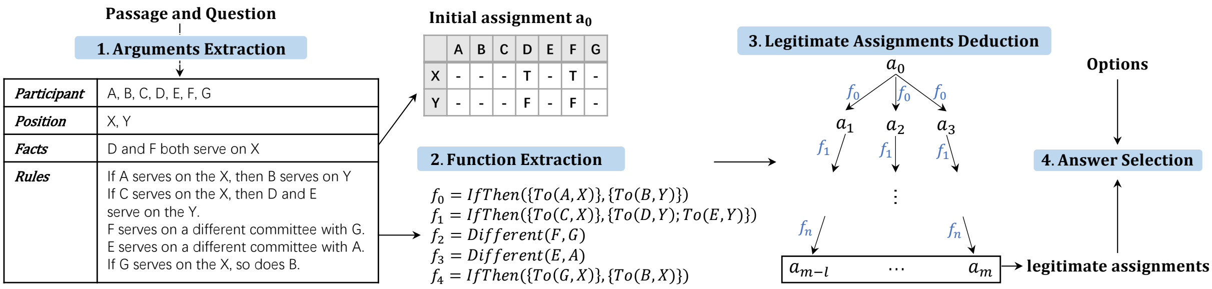

Figure 3: An overview of our approach. The original example is given in Figure 1. It extracts arguments from the context (§ 4.1). Then it extracts constraint functions based on rules (§ 4.3). Afterwards, it conducts deduction to find legitimate assignments (§ 4.4). Lastly, it matches the options and legitimate assignments for prediction (§ 4.5).

Therefore, this task requires the machine to perform explicit complex reasoning, far beyond just understanding the literal clues presented in the text.

4 Approach

We describe how our system, the Analytical Reasoning Machine (ARM), comprehends the knowledge, performs reasoning over the knowledge, and makes conclusions. Figure 3 gives an overview of our approach. Our system operates in four steps: (1) extracting the participants, positions, facts and rules from the passage and the hypothesis of the question (§ 4.1); (2) interpreting rules into a set of logical constraint functions defined in § 4.2, whose arguments are selected from participants and positions (§ 4.3); (3) reasoning with the logical functions and finally generating a group of legitimate assignments (solutions) that satisfy all the rules (§ 4.4); (4) selecting the most plausible option by matching the legitimate assignments and options (§ 4.5). ARM sheds a light on the logical-level reasoning procedure for analytical reasoning and each procedure can be further developed for both performance and expandability.

4.1 Arguments Extraction

In order to understand the context and formalize the problem, the first step is to extract **the participants, positions, facts and rules expressed in natural language** from the passage and hypothesis of the question. An **assignment** represents a solution that assigns participants to positions. An assignment of participants is represented as a table, whose rows and columns represent participants and positions, respectively. Each grid represents whether a participant is assigned to a position, and has the value of three possible states: $(True, False, Unknown)$. The rules describe the constraints of assignments while the facts describe certain assignments. Therefore, we take the sentences that mention specific assignments (e.g., *A on X*) as facts and the other sentences as rules. Facts represent initial assignments to the grids of the assignment table and the default state is noted with *Unknown*. We take the example in Figure 1 as a running example to show the extracted participants, positions, facts and rules from the context.

Specifically, we extract the entities from the leading sentence of the passage with a neural Named Entity Recognition (NER) model (Peters et al., 2017) and group the extracted entities into participants or positions. We parse groups of entities that appear together in the leading sentence of the passage as groups of participants or positions, where participants always appear before positions. For the ordering game, positions can not be directly extracted, so we take them as the order (e.g., *first*, *second*) of participants.

4.2 Constraint Function Definition

We introduce a set of predefined logical functions, which encode constraints expressed in the literal rules and check if an assignment satisfies these constraints. These functions are the foundation of the reasoning process.

The logical functions include three basic types: (1) **relational function**; (2) **compositional function**; (3) **counting function**. A fragment of the predefined functions is shown in Table 2. A function consists of arguments and a executor to check whether an assignment satisfies the constraint function. The detailed definition of each function is listed in Appendix B.

Relational Function The relational functions represent the constraints of the relationship between two participants or a participant and a position. The arguments of relational function involve participant or position. For example, the function

Before$(Ginny, Fernando)$ indicates that *Ginny* should be in the position before *Fernando* in the ordering game. *To*(A, X) indicates that participant A should be assigned to position X.

Compositional Function A compositional function expresses the relationship between two sets of functions, like the conditional rule (*if-then* rule) and the *if-and-only-if* rule. The arguments of compositional functions involve two sets of subfunctions. For example, the rule "*If A serves on the X, then B serves on the Y.*" should be expressed as *IfThen*$(\{To(A, X)\}, \{To(B, Y)\})$.

Counting Function The counting functions focus on the calculation problem of participants under specific constraints. The arguments of counting functions involve a participant and a number. For example, *LastPos*$(A, 3)$ checks whether the participant A is assigned to the last 3 positions.

The input of a function executor is an assignment and the output is a *Bool* value indicates whether the assignment satisfies the constraint.

4.3 Function Extraction

Based on the extracted arguments, we parse the rules expressed in natural language into a set of constraint logical functions that can check whether an assignment satisfy the rules.

One straightforward way is to design a symbolic parsing method. We define an API set to include roughly 20 types of functions like *Before*, *After*, *To*, *IfThen* and realize their executors. For each function, we follow NSM (Liang et al., 2016) that uses trigger words to match a potential function. For example, the function *Before* can be triggered by words "*before*" and "*earlier*". All the functions and trigger words are listed in Appendix B. To extract potential arguments from a given rule, we match the participants, positions, and number from the text. If a function is recognized by a trigger word, we select its arguments from all the potential arguments according to their relative positions to the trigger word. The relational and counting functions can be constituted into compositional functions based on grammar patterns. For example, for the grammar pattern "*If P, then Q*", Each function is grouped into the function set F_1 if it occurs in *P*, or the function set F_2 if it occurs in Q. F_1 and F_2 are taken as the arguments of the function *IfThen*.

Furthermore, to handle the uncertain cases and improve the coverage of extracted functions, we build a neural semantic parsing model based on a pre-trained language model RoBERTa (Liu et al., 2019). It takes the sentence and two parsed arguments in the sentence as the input and predicts their potential function type ("*Null*" if no function exists). Specifically, following Xu et al. (2020), we modify the sentence by adding a special token "@" before and after the first argument, and a special token "#" before and after the second argument. Then, we encode the modified sentence X with RoBERTa to obtain contextual representations H = *RoBERTa*(X). for tokens. Afterwards, we take the representation of the first "@" and "#" for classification.

$$f = argmax(\textit{classifier}([H^{@}; H^{\#}])) \quad (1)$$

where [;] denotes concatenation, and the classifier is a linear layer followed by a softmax function. Since there is no annotated data of corresponding logical functions, we need to construct the training data automatically. The training data consist of (1) positive instances: all the *{input: (rule, arguments); label: function}* pairs that extracted by the symbolic parsing method from the training set; (2) negative instances: the same number of instances that have arguments with no function related.

Afterwards, we extract a set of constraint functions with the combination of symbolic and neural parsing methods. These functions are utilized for reasoning process introduced in the following part.

4.4 Legitimate Assignments Deduction

Given the extracted logical constraint functions and the initial assignment table, we conduct reasoning to find the legitimate assignments that satisfy all the constraints. The process is formulated into a tree-based reasoning algorithm. As shown in Figure 4, each node in a tree corresponds to a table assignment and each edge indicates a constraint function. A node v with path $\{e_0, e_1, ..., e_i\}$ from the root indicates that its assignment satisfies constraint functions $\{f_0, f_1, ..., f_i\}$. Suppose we have n constraint functions, we need to find all the leaf nodes with depth n. These leaf nodes satisfy all the functions and thus become legitimate assignments.

Therefore, we introduce how to construct the complete reasoning tree by the following steps:

1) Firstly, we start with the root, which is the certain initial assignment decided by facts. For the function f_0, we generate all possible assignments related to newly added arguments

Type	Function	Args	Description
Relational Functions	*Before/After*	$participant_1$ $participant_2$	Whether $participant_1$ is in the position before/after $participant_2$.
	Same/Different		Whether $participant_1$ is in the same/different position with $participant_2$.
	To	$participant_1$ $position_1$	Whether $participant_1$ is assigned to $position_1$.
Compositional Functions	*IfThen*	function set F_1 function set F_2	If functions in F_1 satisfied, then functions in F_2 satisfied.
Counting Functions	*FirstPos/LastPos*	$participant_1$, number m	Whether $participant_1$ is assigned to the first/last m positions.

Table 2: A fragment of the logical constraint function definition.

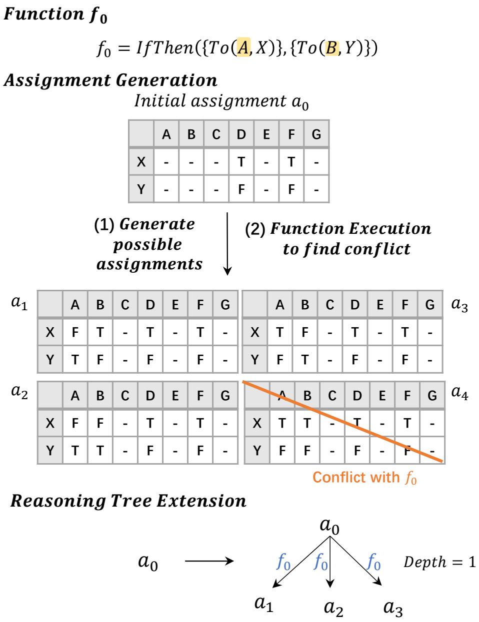

Figure 4: An example of the reasoning process. Newly added participants in f_0 are highlighted. (1) and (2) conducted recursively until $depth = n$. $(T/F/-) = (True/False/Unknown)$

in f_0. As shown in the example in Figure 4, for the function *IfThen*(*To*(A, X), *To*(B, Y)), we generate all possible assignments related to the new participants A and B.

2) We execute f_0 to find all the legitimate assignments that satisfy f_0 as a group of children of the root. In the same example, we keep the assignments that meets *IfThen*(*To*(A, X), *To*(B, Y)).

3) Then we select each child as a new root and select function f_1 for further extension of the reasoning tree.

These processes are recursively conducted until depth n, which means that all the functions are used to construct the reasoning tree. The procedure is summarized into pseudo-code in Appendix A.

It is worth mentioning that although both our algorithm and forward-chaining algorithm deduce new facts based on rules. However, forward-chaining algorithm struggles to do this task because it assumes that all the assignments are already known to the systems while the assignments are always unknown before the deduction steps.

Therefore, this algorithm has advantages of performing explicit interpretable reasoning over the extracted functions and handling uncertain assignments. Moreover, the tree-based manner reduces the computational complexity.

4.5 Answer Selection

Previous steps understand the passage and the question. In this part, we introduce how to analyze the options, and match the options with the deducted legitimate assignments beyond word-level for making a final prediction. Specifically, we can derive two types of information from an option:

1) **Assignment-based option** indicates a table assignment. For example, “*A and C both serve on the X committee*" can be interpreted as a assignment in the table: $\{(A, X) =$ True; $(C, X) =$ True$\}$. For this type, we match the parsed option assignment with all the legitimate assignments and calculate an assignment-based matching score.
2) **Function-based option** indicates an option representing a constraint function, like “*The sedan is serviced earlier in the week than the roadster*", which can be parsed into the function “*Before(sedan, roadster)*". We execute the option-based function on the legitimate assignments to find the satisfiable option and calculate a function-based matching score.

These two types of scores are combined for making a conclusion. The question types and score calculating methods are summarized in the Appendix C.

5 Experiments

We make experiments on the AR-LSAT dataset and evaluate our system with label accuracy. The data split is *(train/dev./test)* $= (1,585/231/230)$ We first compare our system with powerful neural baselines and conduct analysis. Moreover, case study illustrates the reasoning process of our system by an explicit example. Lastly, we make error analysis to point out challenges in this task.

5.1 Model Comparison

Baseline Models We take various powerful neural models, including RNN-based models (i.e., LSTM) and powerful Transformer-based pre-trained language models (i.e, BERT (Devlin et al., 2018), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019), and the recent ALBERT (Lan et al., 2019)) as the baselines of our dataset and investigate their performance. The implementation details of these baselines are given in Appendix D.

Human Performance Since the dataset is based on a test designed for undergraduate students, we select nearly 100 instances in the AR-LSAT dataset and ask 10 undergraduate college students majoring in literature, commerce and law to answer these questions. We take their averaged performance as human performance and report it in Table 3.

Methods	Dev. Acc (%)	Test Acc (%)
Human Performance	-	59.7%
Random Guess	20.0%	20.0%
LSTM	22.5%	20.9%
BERT	23.4%	21.4%
XLNet	23.8%	22.5%
RoBERTa	24.2%	23.1%
ALBERT	24.4%	23.0%
ARM	34.2%	30.9%

Table 3: Performance on the AR-LSAT dataset. Our model is abbreviated as ARM.

Results and Analysis In Table 3, we compare our system (ARM) with baselines and human performance on the development and test set. As shown in Table 1, our model with context understanding and explicit reasoning process significantly outperforms RNN-based models and pre-trained language models with 34.2% accuracy on the development set and 30.9% accuracy on the test set. Results indicate that context understanding and reasoning are essential for this task.

Moreover, we observe that the RNN-based models and pre-trained models struggle to do well on this task, and achieve close performance with random guess. It is also noticed that the performance of both our system and baselines are still far from human performance, leaving significant opportunities for further exploration.

5.2 Model Analysis

In this part, we further analyze the performance and variance of components of our system. To evaluate the performance of arguments extraction, we manually annotate the correct participants and positions in the development set as labels and calculate the accuracy and recall of our condition extraction method and report the results in Table 4. Moreover,

	Acc. (%)	Recall (%)
Participants	96.17	92.88
Positions	84.42	85.79

Table 4: Performance of extraction of participants and positions on the development set.

we eliminate the neural semantic parsing method to evaluate its importance and extract functions by the symbolic parsing method. The results are shown in

Methods	Dev. Acc (%)	Test Acc (%)
ARM	34.2%	30.9%
ARM (w/o neural func.)	32.4%	30.2%

Table 5: Ablation of the the neural semantic parser.

Table 5. Eliminating neural semantic parsing yields no significant compromise in performance. This observation indicates that the neural semantic parsing model can improve performance by improving coverage of the functions and the symbolic parsing method can also provide reliable performance.

5.3 Case Study

We present a case study in Figure 5 to illustrate the reasoning process of our system with interpretable results. Our system extracts correct arguments from the context, and interprets the rules into logical constraint functions. Afterwards, we perform deduction to find legitimate solutions. Lastly, our system matches the options with the legitimate solutions and calculates a score for each option. Option A achieves the highest score because it accords with legitimate assignments. This analysis demonstrates that our system has better explict interpretable reasoning ability.

Passage: A professor must determine the order in which five of her students — Fernando, Ginny, Hakim, Juanita, and Kevin — will perform in an upcoming piano recital. Each student performs one piece, and no two performances overlap. The following constraints apply: Ginny must perform earlier than Fernando. Kevin must perform earlier than Hakim and Juanita. Hakim must perform either immediately before or immediately after Fernando.
Question: If Juanita performs earlier than Ginny, then which one of the following could be true?
Options: (*A*) Fernando performs fourth. √ (*B*) Ginny performs second. (*C*) Hakim performs third. (*D*) Juanita performs third. (*E*) Kevin performs second

Participants & Positions	Fernando, Ginny, Hakim, Juanita, Kevin	first, second, third, fourth, fifth
Rules & Functions	(1) Ginny must perform earlier than Fernando. (2) Kevin must perform earlier than Hakim and Juanita. (3) Hakim must perform either immediately before or immediately after Fernando. (4) Juanita performs earlier than Ginny	(1) *Before (Ginny, Fernando)* (2) *And ({Before (Kevin, Hakim)}, {Before(Kevin, Juanita)})* (3) *Or ({Next (Hakim, Fernando)}, {Last (Hakim, Fernando)})* (4) *Before (Juanita, Ginny)*
Legal Assignments		

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
Fernando	F	F	F	T	F
Ginny	F	F	T	F	F
Hakim	F	F	F	F	T
Juanita	F	T	F	F	F
Kevin	T	F	F	F	F

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
Fernando	F	F	F	F	T
Ginny	F	F	T	F	F
Hakim	F	F	F	T	F
Juanita	F	T	F	F	F
Kevin	T	F	F	F	F

Option Scores (*A*) 1 (*B*) −1 (*C*) −1 (*D*) −1 (*E*) −1

Figure 5: A case study on the AR-LSAT dataset. Our system correctly extracts participants, positions, and rules from the context. Afterwards, it interprets rules into logical functions. After deduction, our system finds legitimate assignments and makes the correct prediction. Rules are highlighted in blue.

5.4 Error Analysis

We randomly select 50 wrongly predicted instances from the dev. set and summarize the error types.

The dominant error type is that some rules with complex semantics are not covered by current constraint logical function set. For example, given a rule "*Each crew member does at least one task during the installation.*" , we should map "*At least*" to function *AtLeastNum*. The second type of errors is caused by failing to extract correct participants or positions by the NER model and predefined matching pattern. The third error type is caused by the lack of basic commonsense knowledge, which is required for understanding the concept in the rules. For example, when a passage mentioned "*Six entertainers should be scheduled at 9:00 A.M., 2:00 P.M., etc*" and the rule is "*Some participants should be scheduled in the morning.*", the system fails to match the *morning* with a specific time zone.

5.5 Discussion

We would like to further highlight important directions to facilitate research on analytical reasoning.

One of the major challenges lies in deep understanding of the knowledge in the context, like parsing the rules into logically equivalent symbolic functions. Deriving machine-understandable functions from natural language is an essential step towards deeper understanding and reasoning. Although supervised semantic parsing has achieved promising progress in recent years, obtaining complete human-annotated logical functions is impractical for this task. Therefore, further study can focus on function extraction with limited amount of annotated functions.

Furthermore, a better inference engine built upon logical functions is also essential because AR questions require deeper reasoning abilities far beyond just understanding the literal clues. Standard symbolic systems like expert systems can provide explicit reasoning, but they are difficult to deal with uncertainty in data. Although neural-based methods are more flexible at dealing with uncertainty, they still struggle to perform interpretable and explicit reasoning. It is promising to better integrate neural and symbolic systems to improve this task with deeper reasoning ability.

6 Conclusion

In this paper, we study the challenging task of analytical reasoning and introduce a dataset AR-LSAT to facilitate research on analytical reasoning. We analyze the knowledge understanding and reasoning ability required for this task and present a system, Analytical Reasoning Machine (ARM), which can comprehend the knowledge, including participants, facts and rules mentioned in the context and extract logically equivalent logical functions from the rules. Afterwards, it performs deep reasoning to find all the legitimate solutions to the problem posed and finally makes a prediction. Experiments show that our system outperforms strong Transformer-based baselines, which indicates that knowledge understanding and deep reasoning is essential for this task. Results show that this task is very challenging for current neural-based models.

References

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen tau Yih, and Yejin Choi. 2019. Abductive commonsense reasoning.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Peter Clark, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Turney, and Daniel Khashabi. 2016. Combining retrieval, statistics, and inference to answer elementary science questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. pages 177–190.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with lstm.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019a. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. *arXiv preprint arXiv:1909.00277*.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019b. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Naoya Inoue, Pontus Stenetorp, and Kentaro Inui. 2020. R4C: A benchmark for evaluating RC systems to get the right answer for the right reason. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6740–6750, Online. Association for Computational Linguistics.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTail: A textual entailment dataset from science question answering. In *AAAI*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D Forbus, and Ni Lao. 2016. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. *arXiv preprint arXiv:1611.00020*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Hanmeng Liu, Leyang Cui, Jian Liu, and Yue Zhang. 2020a. Natural language inference in context–investigating contextual reasoning over long texts. *arXiv preprint arXiv:2011.04864*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020b. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *ArXiv*, abs/1910.14599.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of the 12th International Workshop on semantic evaluation*, pages 747–757.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A Smith, and Yejin Choi. 2018. Event2mind: Commonsense inference on events, intents, and reactions. *arXiv preprint arXiv:1805.06939*.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. *arXiv preprint arXiv:1904.01557*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL-HLT*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. olmpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2017. Constructing datasets for multi-hop reading comprehension across documents.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2018. Dialogue natural language inference.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Nan Duan, and Daxin Jiang. 2020. Syntax-enhanced pre-trained model. *arXiv preprint arXiv:2012.14116*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018a. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018b. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. *arXiv preprint arXiv:2002.04326*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. " going on a vacation" takes longer than" going for a walk": A study of temporal commonsense understanding. *arXiv preprint arXiv:1909.03065*.

A Pseudo-code of Legitimate Assignments Deduction

```
Require: A set of constraint functions F = {f_0, f_1, ..., f_n} and an initial assignment a_0
function ConstructTree(node,functions,depth,n)
    if depth == n then:
        return
    end if
    function = functions[depth]
    old_pars = node.participants
    old_assign = node.assignment
    new_pars = find_new_participant(function, old_pars)
    all_assign = gen_all_assign(old_assign, new_pars)
    satisfied = find_satisfied(all_assign, function)
    depth = depth+1
    children = update_notes(node, satisfied, new_pars)
    for child in children do
        ConstructTree(child, functions, depth, n)
    end for
end function
root = Node(a_0)
depth = 0
n = length of F
complete_tree = ConstructTree(root, F, depth, n)
legitimate = nodes in complete_tree with depth n
return legitimate =0
```

B Function Definition

In this part, we present the detailed description and trigger words for each logical constraint functions in Table 8.

C Question Type

In this part, we list common question types in the AR-LSAT datasets and their ratio in Table 6 and give examples in Table 7. We further introduce how we calculate a score for dominant question type with a group of legitimate assignments.

1) **Must be true/false**: this question type needs to select answer that must be true in all the assignments. We match all the assignments with the option. If one option accords/conflicts with one assignment, the single matching score will be 1/-1, otherwise the score will be 0. We then calculate the sum of all the matching scores as the final score.

2) **Could be true/false**: this question type needs to select answer that could be true in one of the legitimate assignments. We match all the assignments with the option. If one option accords/conflicts with one assignment, the single matching score will be 1/-1, otherwise the score will be 0. We then calculate the maximum matching scores as the final score. The *Acceptable solution* question type also use this method to calculate score.

3) **Maximum number of participants in a position**: this question type needs to calculate the maximum possible number of participants in a specified position (group). We calculate the maximum number of participants in all the legetimate assignments and calculate the absolute difference with the number in the option as the final score.

4) **Find the earliest position of a participant**: this question type needs to calculate the earliest possible position of a specific participant. We calculate the index of the earliest position of the participant in all the legitimate assignments and calculate the absolute difference with the number in the option as the final score.

5) **Count the number of possible positions that a participant can be assigned in**: for this question type, we count all the non-repetitive assignments of the specific participant and calculate the absolute difference with the number in the option as the final score.

D Baseline Models

D.1 Descriptions

- **LSTM** (Gers et al., 1999) is a classical RNN-based model. We apply Bi-LSTM with GloVE (Pennington et al., 2014) embedding.

- **BERT** (Devlin et al., 2018) is a transformer-based model pre-trained on BooksCorpus and Wikipedia with two unsupervised learning task: Masked LM and Nest Sentence Prediction.

- **XLNet** (Yang et al., 2019) is also a transformer-based model, pre-trained on BooksCorpus, Wikipedia, Giga5, ClueWeb 2012-B and Common Crawl with Permutation Language Modeling.

- **RoBERTa** (Liu et al., 2019) is a transformer-based model with the same model structure as BERT but trained on a larger corpus and on a different training setting.

- **ALBERT** (Lan et al., 2019) is a most recent transformer-based pre-trained model. ALBERT uses parameter-reduction techniques that support large-scale configurations.

Question Type	Description
Acceptable solution (15.6%)	identify a feasible solution that can satisfy all the rules
Complete list (3.5%)	identify a complete and accurate list of participants under given condition
Could be true/false (26.8%)	select answer that could be true/false under given condition
Must be true/false (26.4%)	select answer that must be true/false under given condition
Negation (14.7%)	questions that contain negation
Substitution (4.3%)	identify a new rule that can substitute one of the old rules for the desiring result
Condition for determined solution (3.5%)	identify a new rule so that the feasible solution is determined
Calculation (3%)	calculate possible participants in a group
Earliest/latest position (1.3%)	identify the earliest/latest position that a specific participant can be assigned to
Maximum/minimum members (1.3%)	identify the possible maximum/minimum number of participants in a specific group

Table 6: The ratio and description of each question type in the test set of the AR-LSAT dataset.

Question Type	Example
Acceptable solution	Which one of the following could be the schedule of the students' reports?
Complete list	Which one of the following could be a complete and accurate list of the books placed on the bottom shelf?
Could be true/false with condition	If Himalayans are not featured on day 7. which one of the following could be true?
Must be true/false with condition	If Theresa tests G on the second day. then which one of the following must be true?
Negation	P CANNOT be performed at?
Substitution	Which one of the following. if substituted for the condition that Waite's audition must take place earlier than the two recorded auditions. would have the same effect in determining the order of the auditions?
Condition for unique solution	The assignment of parking spaces to each of the new employees is fully and uniquely determined if which one of the following is true?
Calculation	How many of the students are there who could be the one assigned to 1921?
Earliest/latest position	If Zircon performs in an earlier slot than Yardsign. which one of the following is the earliest slot in which Wellspring could perform?
Maximum/minimum members	What is the minimum number of solos in which Wayne performs a traditional piece?

Table 7: The examples of question types in the AR-LSAT dataset.

D.2 Implementation Details

For all the baselines, we employ cross-entropy loss as the loss function and select AdamW as the optimizer for model training/ fine-tuning. These baselines add a simple classification layer on the top of them and take the the last hidden state as the input. For all the Transformer-based models, we employ base model as the backbone.

Type	Function	Arguments	Description	Trigger Words
Relational Functions	Before	participant 1 participant 2	whether participant 1 is in the position before participant 2	before, above, precede, earlier
	After		whether participant 1 is in the position after participant 2	after, larger, higher bigger, older
	Last		whether participant 1 is in the last position of participant 2	immediately before, last
	Next		whether participant 1 is next to participant 2	immediately after, next
	Adjacent		whether participant 1 is neighboring to participant 2	neighboring, adjacent
	Different		whether participant 1 in the different position with participant 2	different
	Same		whether the first participant in the same position with the second participant	same, also
	BeforeEqual		whether participant 1 before or equals to the position of participant 2	no later
	AfterEqual		whether participant 1 after or equals to the position of participant 2	no earlier
	To	participant position	Whether the participant is assigned to the position	to, on, give, in
Compos. Functions	IfThen	function set 1 function set 2	If rules in rule set 1 satisfied, then rules in rule set 2 satisfied	If... then, If ... , ...
	IFF		Rules in rule set 1 satisfied if and only if rules in rule set 2 satisfied	if and only if
	And		Rules in rule set 1 satisfied and rules in the rule set 2 satisfied	and
	Or		Rules in rule set 1 satisfied or rules in rule set 2 satisfied	or
	Unless		Rules in rule set 1 satisfied unless rules in rule set 2 satisfied	unless
	Neither		Neither rules in rule set 1 satisfied nor rules in rule set 2 satisfied	Neither ... nor ...
Counting Functions	FirstPos	participant number	Whether the participant is in the last (number) positions	one of the last (number)
	LastPos		Whether the participant is in the first (number) positions	one of the first (number)

Table 8: Detailed function descriptions and corresponding trigger words