Abstractions for Narrative Comprehension Tasks

Yi-Chun Chen Justus Robertson Arnav Jhala

Department of Computer Science
North Carolina State University
Raleigh, NC 27695
{ ychen74 | jjrobert | ahjhala }@ncsu.edu

Abstract

This paper presents ongoing work in investigating the scale at which semantic abstractions are useful in intelligent reasoning about narrative. One method of evaluating narrative reasoning is to use comprehension tests on stories based on question-answering. Recent advances in language processing have led to promising results in general question-answering. However, current systems fail to accurately answer questions when information is not explicitly mentioned in the input story. Specifically, we are interested in testing whether corpus-based deep learning methods can be extended with classical logic-based approaches to draw inferences beyond ones that are explicitly mentioned in sentences on the corpus. This paper describes a preliminary re-implementation of current methods on the bAbI corpus for question-answering and then presents an algorithm for reasoning about missing information in the input by removing sentences from the corpus.

1 Introduction

Narrative comprehension in humans has long been tested at all levels starting with children to adults with question-answering tasks on short passages of text. These tests have also been used as a way to assess advances in Natural Language Processing and Understanding. Approaches to question-answering have ranged from rule-based systems [RT00], where rules were hand-crafted for different questions types (Who, What, When, Where, Why, and How), to more recently deep learning approaches.

In this paper, we revisit question-answering as a way to assess narrative comprehension. We begin to look at the scale at which semantic abstraction is useful in developing systems that not only answer questions correctly but also can provide reasoning and explanation abilities. As a first step in this process, we present a re-implementation of a current question-answering framework and corpus bAbI [WBC^{+}15]. We then modify bAbI input to explicitly remove information from the input story and show that mapping the input to semantic frames that are represented as PDDL operators allows the model to run classical planning algorithms to infer missing information, such as possible bindings of location variables, constraints on variable bindings for elimination of responses, and hypothesis generation of possible events in the story that could be useful to respond with multiple possible answers when one is not clearly specified in the input.

In: H. Wu, M. Si, A. Jhala (eds.): Proceedings of the Joint Workshop on Intelligent Narrative Technologies and Workshop on Intelligent Cinematography and Editing, Edmonton, Canada, 11-2018, published at http://ceur-ws.org

2 Related Work

Reading comprehension and question answering tasks have a long history in AI, which reaches back to the formative years of the field. Many early approaches [GJWCL61, Sho74, HSSS78] were information retrieval systems which translated natural language questions into logical queries that could be run on a curated database, like a collection of Major League Baseball statistics. Later work [Leh77, SH00] shifted into the realm of natural language processing by building specific knowledge representations and inference techniques to answer user-generated questions about natural language stories, like news articles. Most modern approaches are data-driven and allow many different machine learning architectures to compete on a shared training, validation, and test corpus engineered for a particular purpose. The first of these approaches, Deep Read [HLBB99], used a corpus of 3rd to 6th grade reading comprehension test stories paired with short-answer questions and an answer key. The corpus contained 120 total stories split into 60 development and 60 test stories. Deep Read used a simple bag-of-words technique to retrieve a sentence from the original story in answer to each question and was correct 30-40% of the time on the test set. These initial results were shortly improved on by subsequent work in the field that used different models and approaches on the same corpus of stories [NTK00, RT00].

The current generation of reading comprehension datasets began with MCTest [RBR13], a dataset of 500 fictional stories paired with multiple-choice questions generated and curated by crowd-sourced workers. Following MCTest, many specialized large-scale datasets were created to challenge and steer reading comprehension research. Datasets have been built with text sources pulled from Wikipedia articles [YYM15, RZLL16], publicly available news stories [HKG+15], children's literature [HBCW16], short informational excerpts paired with trivia-questions [JCWZ17], and English exams for middle and high school students [LXL+17]. Some datasets introduce special variations, like QAngaroo [WSR18], which requires models synthesize information from several related documents. These large datasets have spurred competition, progress, and development among deep learning architectures [SKFH17, HZSC18, CG18], which has prompted updates to the datasets themselves [RJL18].

While deep learning approaches are making rapid progress on large natural language QA datasets, many commonsense reasoning tasks have yet to be solved in the context of reading comprehension. A popular open commonsense reasoning test is The Winograd Schema Challenge [LDM11], a collection of short sentences with a single ambiguous word that can be resolve in one of two ways. These ambiguities can easily be resolved correctly by humans based on context but at the first Winograd Schema Challenge competition at IJCAI 2016 [DMOJ17] the best machine score was 58%. A test similar to the Schema Challenge is the Story Cloze Test [MCH+16] where the participant is given a four sentence story or context. After the context, the participant is given two additional sentences where one is a logical conclusion to the story, but the other is not. The score of machine participants has been rising since the test's introduction, but the best score is currently 75% [MRL+17]. In this paper, we explore a logic-based [WFGH06] approach to narrative modeling, comprehension, and question answering. We use a synthetic QA corpus called bAbI [WBC+15] as a testbed for our approach. bAbI's tests are modular and targeted at specific abilities needed for commonsense reasoning in reading comprehension like deduction, induction, path finding, and positional reasoning. We plan to extend and modify these test cases in order to scaffold our approach up to tackle new types of problems.

3 Initial Model

Reading comprehension is the ability of the mind to process and understand text, and then bridge the meaning with experienced knowledge. These underlying information processing systems allow humans to comprehend natural language. The processes interpret incoming information to make suitable responses within the context of questions or situations. In this research, we used a model to imitate the information processing system behind reading comprehension to let a computer accomplish reading comprehension tasks.

In order to examine how our model performs, we ask questions about content with or without explicit information, and let the model give possible answers. Like the key stages that are suggested by many information processing theories, our model follows a sequential method of input-processing-output. Information gathered from the text (input), is stored and processed by the core of the model for later use (processing). When questions are asked, the model decides what it is going to do with the information and how to give suitable responses (output).

3.1 Domain knowledge

When using question-answering as the testing method for reading comprehension task, we frame the task as a 3 elements set: $\{P, Q, An\}$. Where the P denotes the passage which composed by a set of sentence $S = \{s_1, s_2, \ldots.\}$. The Q represents a set of questions regarding the passage content. By mapping the element in Q through the answer function $an_k = answer\_function(q_k)$, we map the questions to a set of possible answers An.

3.2 Example Input Text

We frame our technical discussion with a simple example text that will be used as a running examples through the paper. The text is a three sentence description of a character named Daniel moving around and taking action in a story world.

The sentences are numbered 1 through 4 and end with a question about Daniel's location:

> 1. Daniel journeyed to the office. 2. Daniel grabbed the football. 3. Daniel left the milk. 4. Where is Daniel?

This example text is a simplified bAbI [WBC$^+$15] QA framework test. We will use the Daniel example to ground our technical discussion before introducing the full bAbI framework.

3.3 Input Stage

In this stage, our model has a process to read and encode information from raw text. Instead of storing whole text, at the first step, we use a logical knowledge representation framework as a tool to extract explicit concepts as well as relations in text. Inspired by a knowledge representation framework Rensa [Har17], when our model analyzes the text, the passage information was framed as different relation in Table 1.

In narrative representations, the existence of concepts and how they are related to one another are required in some shape or form. The text base and the story world that is composed by the text encodes these information units. To get these concepts and further use them, we need a representation which can extract the encodings from input data. We take knowledge representation frameworks, like Story Intention Graphs [EM07] and Rensa [Har17], as inspiration for a hand-encoded knowledge representation. We use unary relations to model concepts. A unary relation is a complex concept that includes left-hand (l), right-hand (r), binary relations, and attributes for an information unit, and is written $\phi(x)$. Therefore, each assertion can be written as:

$$\{l(\lambda), relation(\gamma), r(\rho), \phi_1(x_1), \phi_2(x_2), ...\phi_n(x_n)\} \tag{1}$$

The idea of left-hand and right-hand concepts represents a name and value pair, where the name is the label that expresses the subject of the assertions whereas the value could be an array of attributes or another assertion to support the nested relation structure. The name and the value are denoted by λ and ρ respectively. The symbol γ represents the relation between the left and right part. Following the framework, we implemented some classic relations and variable relations in the context of the Daniel example passage in Table 1. When input the raw text, we encode the data into assertions. By this step, the entities in the described world and their perceivable actions and attributes are extracted. These concepts are then stored for later use.

relation	definition	example
is_a / type_of	A hyponym λ is a type of hypernym ρ (class inclusion)	is_a(Daniel, entity)
action	λ perform action ρ	action(Daniel, journeyed)
action_at	λ is located at ρ	action_at(Daniel, office)
has_property	λ is ρ	has_property(garden, uncertain)

Table 1: Relations used by our system in the context of the Daniel example text.

In each sentence, the information can be divided to entity E and action A parts where the entities are nouns in this sentence and the actions are verbs. Therefore s_k denotes a sentence where $s_k = \{E_k, A_k\}$. To reason about the information in semantic level of the action, we first integrate hand-crafted rules with the *predicates* and *thematic roles* of verbs from Verb-Net [KKRP08] [1]. Because the verbs in bAbI tasks are limited, we currently

[1] The Verb-Net project is in https://verbs.colorado.edu/verbnet/

are able to separate the verbs into three categories: $CONNECT$, $SEPARATE$, and $MOVE$. If the predicates of a verb include a predicate that shows changing on locations, the verb will be categorized into the $MOVE$ group. If the predicates of a verb contain the relation between source and theme, it will be categorized to the $CONNECT$ group. And if the predicates of a verb include the relation between theme and destination or an end location, it will be considered to the $SEPARATE$ group.

After the action in the sentence is put into a group, we process the entities and relations in the sentence. Besides the predicates and thematic roles of a verb, Verb-Net also provides *syntax frames* for each verb. When processing entities, we match syntax frames with sentences to fill entities into thematic roles. In order to better reason over the relations between entities, they are divided into three different sets. According to the characteristics of the entities, the sets are characters C, locations L, and other objects O.

When we process the entities we categorize them by their characteristics. When a name matches some linguistic rule such as using a capital letter, or is a reference to the subject of a clause, and display signs of animacy such as taking action, the system considers the entity as a character. When we say an entity is capable to conduct actions which means the entity can be filled into the Agent role in verb frames in the sentence, or the entities is exist in certain external resources such as baby's name database, it will be categorize to the character set $C = \{c_k \in E \; and \; isActor(c_k) = True\}$. Similarly, if an entity is linked by prepositions of places, or it can match the Location role in verb frames, it will be categorized as an element in location set $L = \{l_k \in E \; and \; isLocaiton(l_k) = True\}$. For the rest of the entities, if they are the object of the action and is not categorized to character set, they will be put into object set $O = \{o_k \in E \; and \; not(C \cup L)\}$. In these steps, we present the link between groups and actions, sets and entities by the $type\_of$ and is_a relations. For example, $is\_a(Daniel, entity)$, $is\_a(Daniel, Character)$, $type\_of(grab, CONNECT)$. This process is described in Algorithm 1.

Algorithm 1 Reason Action and Entities

```
$S \leftarrow$ input sentence
$[E, A] \leftarrow$ ProcessSentence($S$)
categorize all $a \in A$ to $\{CONNECT, SEPARATE, MOVE\}$ groups
create $type\_of(a, actiongroup)$ relations
while exist $e$ in $E$ is unchecked do
    pick an $e$ arbitrarily
    if $isActor(e) == True$ then
        put $e$ in $\{Character\}$
    else if $isLocation(e) == True$ then
        put $e$ in $\{Location\}$
    else
        put $e$ in $\{Object\}$
    mark $e$ as checked
    create $is\_a(e, \{Character, Location, Object\})$
```

3.4 Processing and Storage Stage

For the information pair of a sentence i C_i, A_i, O_i, we already construct the relations between entities and between entity and actions. By matching the syntax frame, we are also able to create the relation that $action(c_k \in C_i, ak \in A_i)$ where combines the the character c_k and the corresponding action a_k. As a next step, we start to process the relation between actions and their effects. If the action belongs to CONNECT group, we will match elements in O_i and create $has\_possession(c_k, o_k \in O_i)$ relations to link the own-ship between objects and the characters, and this change will reflect to the change of world states. When the action belongs to the SEPARATE group, the $has\_possession(c_k, o_k \in O_i)$ relation will be removed to indicate that the own-ship not hold anymore and our model will updates the world states accordingly. The last case is the MOVE category, when the action belongs to this category, the location information is link to both actions and characters, we get $action(c_k, a_k)$ and $action\_at(c_k, l_k \in L_i)$ relation pairs, the change will also be reflected to world states. Moreover, for each object that links to the character, $has\_possession(c_k, o_k \in O_i)$, their location information will be updated accordingly.

This stage includes adding information to the mental schema and encoding it. As described above, we first analyze the basic relations that inherited from the input stage and then use a rule-based method to bridge them with the knowledge base to figure out the effects that new incoming information applies to story world

Algorithm 2 Reason Action and Effects

```
if $a \in CONNECT$ then
    add $has\_possession(c, o)$ relation
else if $a \in SEPARATE$ then
    remove $has\_possession(c, o)$ relation
else if $a \in MOVE$ then
    update $action_at(c, l)$ and $at(c, l)$ relation
    for all $has\_possession(c, o)$ do
        update $at(o, l)$
update world states
```

states. The relations form the assertions about entities. To build a logical model with the assertions, we use the planning language PDDL [MGH$^+$98] over first-order logic with constant symbols called *objects*, relation symbols called *predicates*, and variable symbols. We start with a set of PDDL operators $\mathcal{O}$ where each operator $o = \langle l_o, p_o, e_o \rangle \in \mathcal{O}$ consists of a unique name or label l, a conjunctive set of first-order literal *preconditions* p, and a conjunctive set of first-order literal *effects* e. Preconditions specify what must be true in the story world for an action to take place and effects specify how the world is updated by the action. Operators have parameters that can be *grounded* by substituting concrete PDDL objects, which represent story world characters, things, and locations for parameter variables. Table 2 shows the parameters, preconditions, and effects of an example PDDL operator for moving a character from one location to another. The operator is ground with objects from our example story.

Algorithm 3 Sentence Processing

```
$S \leftarrow$ input sentence
use speech tagger to get  $nouns$, $verb$  of the sentence
$E \leftarrow$ nouns
$a \leftarrow$ verb of the sentence
return $[E, a]$
```

During the process that interprets actions, effects of actions, and adds preconditions to our model's knowledge base, because the effects are often unstated, background knowledge is often required to reason about the changed world. When applying our model to test sets in experiments, we set some reference rules for actions that appear in the text and update the changes of world states accordingly. In this stage, after processing the input assertions from last stage, our model encodes the changing of world states, conditions, or actions into PDDL format. We represent those result by PDDL-like fluents similar to those used in a planning problem. Once the comprehend world is not continuous, we could insert planning methods to fill in the gaps between the changing of states.

Parameters	Ground Parameters
`?mover ?newlocation ?oldlocation`	`Daniel office kitchen`
Preconditions	**Ground Preconditions**
`(at ?mover ?oldlocation)`	`(at Daniel kitchen)`
`¬(at ?mover ?newlocation)`	`¬(at Daniel office)`
Effects	**Ground Effects**
`(at ?mover ?newlocation)`	`(at Daniel office)`
`¬(at ?mover ?oldlocation)`	`¬(at Daniel kitchen)`

Table 2: Example ground and unground parameters, preconditions, and effects for a PDDL operator that describes how a character moves from one place to another. The parameters are ground using the example story.

3.5 Output Stage

In this stage, the model prepares an appropriate response to outside stimuli. The stimuli are questions that relate to text content and the response should be the answer or possible answers generated from the knowledge base that is constructed by our previous comprehension process. To give answers to the questions, we analyze the them through the processes in the previous two stages. Our model extracts information to make the question be assigned to a specific category. Like the process for input text, we first transform the question into assertions, and then interpret the assertion to get the subject of the question as well as the question type. Example question types are yes/no question, where, or who. This step not only analyzes the question content, but also prepares for generating answers.

For certain questions, our model provides templates to create answers. For other questions, our model gives the most related information which matches the question topic. When our model is exploring its knowledge base for answers, it can encounter different situations. The easiest situation is when the answer can be directly queued from existing world states. However, if the answer doesn't appear in existing world states because the information is implicit or missing our model must process to explore answers. One processing method is to seek other subjects which have a relation with the question subjects in current world states, then use the information to fill out missing relations to produce possible answers. Another process is to trace back to previous world states in order to look up information.

3.5.1 Planning for Missing Information

When our model gets a question regarding missing information, it will try to find a reference entity and plan for the missing part. For example, the question "Where is Daniel?" asks for location information about the character Daniel. If we cannot find the answer from current world states, but the question subject has relations with other entities such as $has\_possession(Daniel, football)$, our model will mark the entity as a reference. For example, our system considers the states about the reference it can infer possible answers from, like $at(Daniel, unknown)$ but $has\_possession(Daniel, football)$ and $at(football, garden)$, then gives the possible answer should be "garden". It also traces back to previous world states where the last time it know about the question subject (Daniel), and with the possible answer $at(Daniel, garden)$, it uses the following planning procedure to figure out the possible missing information. In this example, our system will set previous state $at(Daniel, unknown)$ as the initial state and the possible answer $at(Daniel, garden)$ as the goal state. After that, the model will assign previous world states as preconditions. By matching the preconditions, the system gets a series of actions, and then according to the effects of the actions in the action set, our system choose a possible missing action.

Algorithm 4 Planning

```
$possible\_info$ = []
$initial\_state$ ← previous state about the question subject
$current\_state$ ← $initial\_state$
$goal\_state$ ← the possible answer
$precondition\_set$ ← previous world states
while $current\_state$ not match $goal\_state$ do
    if $precondition\_set$ matches elements in action groups then
        add to $doable\_set$
    if the effect of any $a \in doable_set$ matches $goal\_state$ then
        update $possible\_info$
        $current\_state$ ← effect of $a$
    else
        pick an $a$
        update $possible\_info$
        $current\_state$ ← effect of $a$
        update $precondition\_set$
return $possible\_info$
```

3.6 Performance on bAbI Tasks

The bAbI task is a proxy task which evaluates reading comprehension via question answering. It measures a language understanding system in different aspects of abilities, including whether the system can answer questions via chaining facts, simple induction, deduction, and so on. In our experiments, we use the task set to identify successful and insufficient parts of our model. In our experiments, we get almost 100% correctness in the following tasks sets. We tested tasks 1, 2, 3, 6, 8, and 10. Tasks 1, 2 and 3 provide test sets that answer

Task ID	Task 1	Task 2	Task 3	Task 6	Task 8	Task 10	Task 15	Task 16
MemNet PE LS RN	0.5	8.9	41.4	7.7	12.9	18.9	0.0	0.6
MemNet PE LS RN JOINT	0.1	16.6	26.3	2.8	10.1	6.5	0.3	2.1
Our Model	0.0	0.0	0.001	0.004	0.0	0.0	0.0	0.0

Table 3: The table of error rate (%) on bAbI taks 1, 2, 3, 6, 8, and 10. Comparison with the performance of MemNet with different configuration: 3 hops, position encoding (PE), linear start training (LS), random noise (RN) and adjacent weight tying. The benchmark result come from [WBC^{+}15][3]

questions about single supporting facts, among other facts or questions about the combination of information from sentences. In the simplest case, the task has questions like, "Mary traveled to the office. Where is Mary?" which only asks about information that is provided directly. When the task tests the understanding of two or more facts, the questions require references of other information to get information regarding the question subject. For example, "John is in the playground. John picked up the football. Where is the football?" The question subject is the football, but to answer this question information about John is also needed.

For the simplest type of tasks, our model can directly get an answer from its knowledge base. Because the input sentences already contain explicit information regarding answers, our model can link the effect of actions to world states when processing input sentences. Therefore, the needed information is accessible in the current world states. When answering the questions about a combination of information, our model first queries the information about question subjects from its knowledge base, and then finds the reference object from the queried world states. From the information of the reference object, the question can be answered correctly. Task 6 is very similar to task 1, it asks questions about supporting facts. But it aims to test the ability of a model in answering true/false type questions. Such as "Is John in the playground?" Our model can generate these kind of answers correctly by placing related information into a right category of answer framework. Task 8 requires models to generate a list that is composed by a set of single word answers. A question example is "What is Daniel holding?" Our model answers these questions by searching the states of objects in the described world. Task 10 gives possibility statement instead of facts. Our model preserves uncertain states while processing input sentences in the very beginning, so we can answer the questions. Task 15 and Task 16 test basic deduction and induction, respectively. In Task 15, sentences describe entities with some characteristics. After each characteristic, sentences give a subject that inherits properties from the entity, and then asks questions about the characteristics. Task 16 gives different properties of subjects in sentences. It uses another subject as a question subject and asks if the question subject has some of the properties with the described subject. In our processing stage these assertions are connected through the same subjects and our model stores the relations as states of subjects, the is_a relation helps us to keep the information that be used in deduction or induction. Therefore, our model can also find answers from its knowledge base.

For these tasks, we compare our result with the MemNet which is cited by bAbI paper [WBC^{+}15] in Table 3.

4 Extended Task Framework

Although bAbI tasks cover many aspects of evaluations about understanding models, they give explicit information in each task. In addition to answering questions about described things, we are also interested in whether the model can answer questions with implicit information. When people read a paragraph of text, like a story, changing world states are not always clearly stated. We propose a task framework that modifies the bAbI task to test an aspect of the ability of understanding models. The extended task removes information from the original text in order to test whether a model can use known information to infer the implicit part. The following are examples of the extended task. Here is an example of the original bAbI task, Two Supporting Facts:

> 1. Mary moved to the bathroom. 2. Mary got the football there. 3. John went to the kitchen. 4. Mary went back to the kitchen. 5. Mary dropped the football. 6. John got the football there. 7. Where is the football? Kitchen 4 5

In this case, we can see that the answer to the question is the kitchen because the question subject, the football, was mentioned in sentences 4 and 5. To create an inference task, our modification will be:

> 1. Mary moved to the bathroom. 2. Mary got the football there. 3. 4. Mary went back to the kitchen. 5. Mary dropped the football. 6. John got the football there. 7. Where is John? Kitchen 5 6

We consider items that show up together in a sentence with the question subject. These guarantee there exist certain relations between the question subject and things in the world. Then we remove the information directly related to the question to create the inference part. In this example, because the question asks about location, we remove the sentence which gives location information. To answer these questions, the models must determine information from the relation between the two subjects. For this example, we can answer that John is in the kitchen because he got the football, and the football was dropped in the kitchen. With the same original text, and following the rule we descried above, another modification could be:

> 1. Mary moved to the bathroom. 2. Mary got the football there. 3. John went to the kitchen. 4. 5. Mary dropped the football. 6. John got the football there. 7. Where is Mary? Kitchen 3 5 6

Similarly, the question subject Mary and a described item, the football, are in the same sentence. Also, the location information of the question subject is removed. We can answer this question because Mary dropped the football somewhere, but John got the football in his location, so we know that Mary should be at the same place with John. These modified rules test not only the ability to infer unstated information from given sentences, but also evaluate the ability to understand implications like if a subject can conduct actions on some objects that they must be in the same place. Our model solves the extended task through the following process. In the former modification, the answer searching process starts from the initial state that John is at an unknown location and the football is in the kitchen, and the goal state would be that John is in the same location with the football because he is able to get the football. By searching the possible actions that effect John's state from an unknown place to the kitchen, our model can not only answer the question but also fill back the missing part. In the latter case, the initial state will be that Mary is at an unknown location as well as the football is in the kitchen, and the goal state will be that Mary and the football are in the same location.

5 Conclusions

This work, while preliminary, sets up a platform for further research in question-answering and narrative abstraction. There are several areas of further improvement. We currently use a static set of PDDL operators $\mathcal{O}$ to build our inferred world states from the knowledge representation. In the future, we'd like to dynamically build this set. One avenue is to learn the set by reading stories [CMW13]. Another would be to use a lexicon, like VerbNet [Sch05], to map relations in the knowledge representation to a database of PDDL operators. bABI is one of several corpora that are currently being actively developed within the NLP community. We are already looking at expanding this work to include other datasets, such as SQuAD [RJL18].

References

[CG18] Christopher Clark and Matt Gardner. Simple and Effective Multi-Paragraph Reading Comprehension. In *Meeting of the Association for Computational Linguistics*, pages 845–855, 2018.

[CMW13] Stephen N. Cresswell, Thomas L. McCluskey, and Margaret M. West. Acquiring Planning domain models using LOCM. *The Knowledge Engineering Review*, 28(2):195–213, 2013.

[DMOJ17] Ernest Davis, Leora Morgenstern, and Charles L. Ortiz Jr. The First Winograd Schema Challenge at IJCAI-16. *AI Magazine*, 38(3), 2017.

[EM07] David K. Elson and Kathleen R. McKeown. A Platform for Symbolically Encoding Human Narratives. In *AAAI Fall Symposium on Intelligent Narrative Technologies*, 2007.

[GJWCL61] Bert F. Green Jr., Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: An Automatic Question Answerer. In *Western Joint IRE-AIEE-ACM Computer Conference*, pages 219–224. ACM, 1961.

[Har17] Sarah Harmon. *Narrative Encoding for Computational Reasoning and Adaptation.* PhD thesis, University of California, Santa Cruz, 2017.

[HBCW16] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. In *International Conference on Learning Representations*, 2016.

[HKG$^+$15] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching Machines to Read and Comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

[HLBB99] Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. Deep Read: A Reading Comprehension System. In *Meeting of the Association for Computational Linguistics*, pages 325–332, 1999.

[HSSS78] Gary G. Hendrix, Earl D. Sacerdoti, Daniel Sagalowicz, and Jonathan Slocum. Developing a Natural Language Interface to Complex Data. *Transactions on Database Systems*, 3(2):105–147, 1978.

[HZSC18] Hsin-Yuan Huang, Chenguang Zhu, Yelong Shen, and Weizhu Chen. Fusionnet: Fusing Via Fully-Aware Attention with Application to Machine Comprehension. In *International Conference on Learning Representations*, 2018.

[JCWZ17] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Meeting of the Association for Computational Linguistics*, 2017.

[KKRP08] Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. A Large-scale Classification of English Verbs. *Languge Resources and Evaluation Journal*, 42(1):21–40, 2008.

[LDM11] Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The Winograd Schema Challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47, 2011.

[Leh77] Wendy G. Lehnert. A Conceptual Theory of Question Answering. In *International Joint Conference on Artificial Intelligence*, pages 158–164, 1977.

[LXL$^+$17] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[MCH$^+$16] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James F. Allen. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, 2016.

[MGH$^+$98] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL - The Planning Domain Definition Language. Technical Report CVC TR98003/DCSTR1165, Yale Center for Computational Vision and Control, 1998.

[MRL$^+$17] Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James F. Allen. LSDSem 2017 Shared Task: The Story Cloze Test. In *Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51, 2017.

[NTK00] Hwee Tou Ng, Leong Hwee Teo, and Jennifer Lai Pheng Kwan. A Machine Learning Approach to Answering Questions for Reading Comprehension Tests. In *Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 124–132, 2000.

[RBR13] Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text. In *Conference on Empirical Methods in Natural Language Processing*, pages 193–203, 2013.

[RJL18] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Meeting of the Association for Computational Linguistics*, 2018.

[RT00] Ellen Riloff and Michael Thelen. A Rule-Based Question Answering System for Reading Comprehension Tests. In *Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, pages 13–19, 2000.

[RZLL16] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ Questions for Machine Comprehension of Text. In *Conference on Empirical Methods in Natural Language Processing*, page 2383, 2016.

[Sch05] Karin Kipper Schuler. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon.* PhD thesis, University of Pennsylvania, 2005.

[SH00] Lenhart K. Schubert and Chung Hee Hwang. Episodic Logic Meets Little Red Riding Hood: A Comprehensive, Natural Representation for Language Understanding. *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language*, pages 111–174, 2000.

[Sho74] Edward H. Shortliffe. A Rule-Based Computer Program for Advising Physicians Regarding Antimicrobial Therapy Selection. In *ACM Conference*, pages 739–739, 1974.

[SKFH17] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional Attention Flow for Machine Comprehension. In *International Conference on Learning Representations*, 2017.

[WBC$^+$15] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. *arXiv preprint arXiv:1502.05698*, 2015.

[WFGH06] Ben Wellner, Lisa Ferro, Warren Greiff, and Lynette Hirschman. Reading Comprehension Tests for Computer-Based Understanding Evaluation. *Natural Language Engineering*, 12(4):305–334, 2006.

[WSR18] Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing Datasets for Multi-hop Reading Comprehension Across Documents. *Transactions of the Association for Computational Linguistics*, 6:287–302, 2018.

[YYM15] Yi Yang, Wen-tau Yih, and Christopher Meek. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, 2015.