Combining Retrieval, Statistics, and Inference to Answer Elementary Science Questions

Peter Clark, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Turney
Allen Institute for Artificial Intelligence
{peterc,orene,tushark,ashishs,oyvindt,petert}@allenai.org

Daniel Khashabi
Cognitive Computation Lab (CCG), Univ Illinois, Urbana-Champaign
khashab2@illinois.edu

Abstract

What capabilities are required for an AI system to pass standard 4th Grade Science Tests? Previous work has examined the use of Markov Logic Networks (MLNs) to represent the requisite background knowledge and interpret test questions, but did not improve upon an information retrieval (IR) baseline. In this paper, we describe an alternative approach that operates at three levels of representation and reasoning: information retrieval, corpus statistics, and simple inference over a semi-automatically constructed knowledge base, to achieve substantially improved results. We evaluate the methods on six years of unseen, unedited exam questions from the NY Regents Science Exam (using only non-diagram, multiple choice questions), and show that our overall system's score is 71.3%, an improvement of 23.8% (absolute) over the MLN-based method described in previous work. We conclude with a detailed analysis, illustrating the complementary strengths of each method in the ensemble. Our datasets are being released to enable further research.

Introduction

Our goal is a system that performs well on Elementary Science tests, a grand challenge for AI because of the wide variety of knowledge and reasoning skills required (Clark 2015). Although not itself an application, Fourth Grade test-taking requires question answering (QA) that goes significantly beyond retrieval techniques, yet is simple enough to be accessible. In this sense, it is a simple embodiment of an AI challenge requiring both language and reasoning, suitable as part of a broader Turing Test (Clark and Etzioni 2016), and with many potential applications if solved (e.g., tutoring systems, a scientist's assistant). This goal is hard, for example several Markov Logic Network (MLN) systems aimed at this task were reported by Khot et al. (2015), but the best MLN formulation still did not outperform an IR baseline.

Our approach is to use an ensemble of solvers operating at different levels of representational structure, each capable of answering different genres of question reliably. Consider a question from the NY Regents 4th Grade Science Test:

Fourth graders are planning a roller-skate race. Which surface would be the best for this race? (A) gravel (B) sand (C) blacktop (D) grass

This question might be answered in several ways. There may be a sentence on the Web that happens to state the answer explicitly (e.g., "Blacktop is a good surface for a roller-skating race"), and could be reliably retrieved to support the correct answer ("blacktop"). Alternatively, corpus statistics may reveal that "blacktop" is strongly associated with "roller skating" and "race", indicating that blacktop is the right answer. Finally, we might infer the answer from more general knowledge, e.g., roller-skating requires a smooth surface, and blacktop has a smooth surface, therefore blacktop is good for roller-skating.

Similarly our system, called Aristo, solves non-diagram multiple choice questions using five algorithms ("solvers") operating at different levels of structure. The combination addresses some of the reported problems with the earlier MLN systems. First, any system attempting reasoning over text needs some fallback methods when reasoning fails. We provide this using a baseline information retrieval (IR) solver. Second, some simpler questions can be answered using just statistical associations between questions and answer options, but the MLN systems made no attempt to use that knowledge. We address this using two solvers that exploit pointwise mutual information (PMI) and word embeddings (generating features for an SVM) respectively. Third, for more complex questions, we use a RULE solver that uses soft logical rules extracted from text. We also include a solver that uses a structurally simpler knowledge representation (tables), applied using integer linear programming (ILP). This provides an additional reasoning method that avoids the noise and complexity of the extracted rules. This combination thus covers a spectrum of representational levels, allowing questions to be robustly answered using retrieval, statistics, and inference. Solver scores are combined using logistic regression.

Our contributions are as follows:

- By combining solvers working at different levels of representation, our system Aristo achieves a significantly higher performance (71.3%) than the best previously published method (Khot et al. 2015).
- We carry out ablation studies that quantify the contribution of each method to Aristo, and show that all levels of representation help. Our error analysis indicates the complementary strengths and weaknesses of each method, and

directions for future work.

- We show that the challenge problem itself is a valuable testbed for AI research, and are releasing our datasets (at www.allenai.org) to encourage further research.

Related Work

Question Answering (QA) has been extensively studied in the past few years, but has primarily focused on retrieving answers to short, factoid questions (e.g., "In which year was Bill Clinton born?") by locating answers in databases (Yao and Van Durme 2014; Zou et al. 2014; Fader, Zettlemoyer, and Etzioni 2014) or large document collections (Brill, Dumais, and Banko 2002; Ferrucci et al. 2010; Ko, Nyberg, and Si 2007). In contrast, many science questions do not have answers explicitly stated in text, and require some form of analysis or inference to answer them. While there are good examples of inference-based QA systems (Gunning et al. 2010; Novak 1977), they require questions to be posed in logic or restricted English, and were not applied to natural questions. A few systems have attempted standardized tests, e.g., in geometry (Seo et al. 2014) and mathematics (Hosseini et al. 2014; Kushman et al. 2014), and work well due to their limited domains and stylized wording of questions. Our work investigates a far less constrained domain, and thus utilizes different methods.

Answering questions using an ensemble has been shown to be effective in numerous previous cases, (e.g., Töscher, Jahrer, and Bell (2009)), most famously in IBM's highly successful Watson system (Ferrucci et al. 2010). What is novel here is the nature of the problem being addressed, namely single and multi-sentence science questions whose answers may require statistical or structured reasoning. By instantiating this architecture with modules at different levels of structure, Aristo can both leverage text when an answer is explicitly stated in a corpus, and perform inference to go beyond textual information when it is not. This latter capability allows Aristo to answer questions out of reach of corpus-based methods, without losing the powerful capabilities such methods provide.

Approach

Aristo's overall architecture, shown in Figure 1, consists of five solvers that work in parallel to answer a multiple choice question. The IR solver operates directly on the text. The PMI solver and the SVM solver use statistical data derived from text. The RULE solver and the ILP solver reason with knowledge extracted from text. Each solver assigns confidences to each of the answer options, and a combiner module combines the results together using logistic regression trained on a set of training examples.

Layer 1: Text as Knowledge

The Information Retrieval (IR) Solver

The IR solver searches to see if the question q along with an answer option is explicitly stated in a corpus, and returns the confidence that such a statement was found. For each answer option a_i, it sends $q + a_i$ as a query to a search engine (we use Lucene), and returns the search engine's score for the top retrieved sentence s where s also has at least one non-stopword overlap with q, and at least one with a_i; this ensures s has *some* relevance to both q and a_i. This is repeated for all options a_i to score them all.

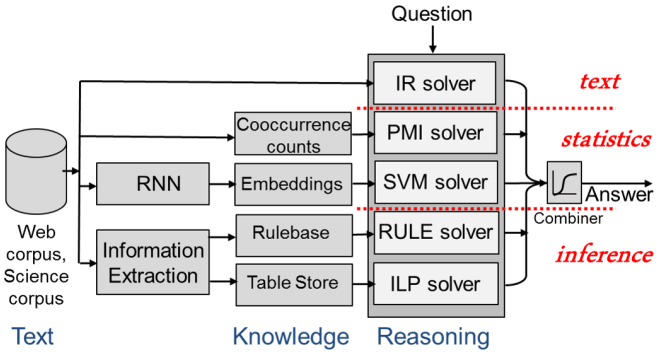

Figure 1: Aristo uses five solvers, each using different types of knowledge, to answer multiple choice questions.

Layer 2: Statistical Knowledge

The IR solver provides a surprisingly strong baseline, but, as we show later, it is clearly limited in at least two ways. First, it requires the answer to a question to be explicitly contained somewhere in the corpus. Second, it requires the wording of that answer to be reasonably similar to that used in the question, so that a retrieval engine will rank it highly.

Aggregations over a corpus provides an alternative, weak source of commonsense knowledge. Consider:

> *A mother hen clucks loudly when danger is near and her chicks quickly gather around her. Which sense helps the chicks receive this warning about danger from their mother? (A) smell (B) taste (C) sight (D) sound*

An IR approach struggles with this wordy question. However, the simple commonsense knowledge that the question appeals to is that a "cluck" is a "sound", or, more weakly, that "cluck" and "sound" are strongly associated. We can use corpus statistics to measure such associations. Although such statistics do not tell us the nature of that association, in many cases knowing the existence of the association provides a strong enough signal for question-answering.

The Pointwise Mutual Information (PMI) solver

The PMI solver formalizes a way of computing and applying such associational knowledge. Given a question q and an answer option a_i, it uses pointwise mutual information (Church and Hanks 1989) to measure the strength of the associations between parts of q and parts of a_i. Given a large corpus C, PMI for two n-grams x and y is defined as:

$$PMI(x, y) = \log \frac{p(x, y)}{p(x)p(y)}$$

Here $p(x, y)$ is the joint probability that x and y occur together in the corpus C, within a certain window of text (we use a 10 word window). The term $p(x)p(y)$, on the other hand, represents the probability with which x and y would

occur together if they were statistically independent. The ratio of $p(x, y)$ to $p(x)p(y)$ is thus the ratio of the observed co-occurrence to the expected co-occurrence. The larger this ratio, the stronger the association between x and y.

We extract unigrams, bigrams, trigrams, and skip-bigrams from the question q and each answer option a_i. We use the SMART stop word list (Salton 1971) to filter the extracted n-grams, but allow trigrams to have a stop word as their middle word. The answer with the largest average PMI, calculated over all pairs of question n-grams and answer option n-grams, is the best guess for the PMI solver.

The Support Vector Machine (SVM) solver

Word association is one statistical approach to question-answering. An alternative is word similarity, specifically how semantically related an answer option a_i is to the question q, where word embeddings are used to represent the lexical semantics.

We use the lexical semantics model and implementation, created by Jansen, Surdeanu, and Clark (2014), to generate domain-appropriate embeddings for a corpus of elementary science text. The embeddings are learned using the recurrent neural network language model (RNNLM) (Mikolov et al. 2010; 2013). Like any language model, a RNNLM estimates the probability of observing a word given the preceding context, but, in this process, it also learns word embeddings into a latent, conceptual space with a fixed number of dimensions. Consequently, related words tend to have vectors that are close to each other in this space. We derive two measures from these vectors. The first is a measure of the overall similarity of the question and answer option, which is computed as the cosine similarity between the two composite vectors of q and a_i. These composite vectors are assembled by summing the vectors for individual words of q (or a_i), and re-normalizing this composite vector to unit length. (This approach to combining vectors is simple and has worked well previously, e.g., (Yih et al. 2013)). The second measure is the average pairwise cosine similarity between each word in q and a_i. This is repeated using four alternative science corpora (two features per corpus), and combined using an SVM ranker trained on a set of multiple choice questions with known answers, to compute a score for each answer option. We use an SVM as it has been previously shown to perform comparably with other algorithms for answer ranking, e.g., (Surdeanu, Ciaramita, and Zaragoza 2011), with mature software available.

Layer 3: Structured Knowledge

While textual and statistical systems perform well, they operate without any deep understanding of the question or domain, and consequently can be misled. This is especially true for questions for which some representation of general truths about the domain and methods for applying them is required. We explore two solvers that attempt to capture and apply this kind of knowledge, in complementary ways.

The RULE Solver

The RULE solver is based on the knowledge representation introduced by Clark et al. (2014), which is in the form of rules expressed in a probabilistic (subset of) first-order logic, and extracted automatically from text. Knowledge acquisition occurs in a two-step process. First, a corpus of science text is parsed and then scanned using a small number of (hand-authored) extraction patterns to identify expressions of seven types of implication in text (*cause, enables, purpose, requirement, condition, part, example*, chosen by manual analysis of questions for the most commonly queried relationships). Although manually created, the patterns are domain-general so in principle would apply to a new domain also. Each pattern maps a syntactic structure to the form (*tuple implication tuple*), for example one pattern maps the textbook sentence *"Some animals grow thick fur to stay warm."* to the structure:

((“Some animals” “grow” “thick fur”) EFFECT
(“Some animals” “stay” “warm”))

Second, the extracted structure – a generic statement – is converted to an implication by applying a default reading of generics, where (*A relation B*) is interpreted as “for all instances of A, there exists a B that is in *relation* to A”, producing a “forall...exists...” implication. For example, the above produces:

// IF An animal grows thick fur
// THEN the animal stays warm
$\forall$ a, g, t isa(a,”Some animals”), isa(g,”grow”),
isa(t,”thick fur”), agent(g,a), object(g,t)
$\rightarrow \exists$ s,w isa(s,”stay”),isa(w,”warm”)
agent(s,a), object(s,w), effect(g,s).

Note that this representation is “semi-formal” as we retain words/phrases in the structures to denote concepts. A simple textual entailment service is used during reasoning to determine the confidence in equality between different text strings, e.g., isa(x,”bear”) entails, with some confidence, isa(x,”animal”).

To apply such knowledge, the question q and an answer option a_i are translated into similar structures using the same natural language processing (NLP) machinery. The reasoner then performs a best-first search to find the lowest cost application of rules to derive the answer option (the conclusion) from the question (the premise). To tolerate incompleteness in the KB, the system mixes logical reasoning and lexical matching to determine the confidence that a rule can be applied, where the confidence is a normalized average of the entailment scores between literals in the question interpretation and the rule's antecedent, and the entailment scores between words in the question and words in the rule's antecedent. This allows rules to be applied even if their antecedents are only partially satisfied, providing a degree of robustness. The solver returns the (inverted) cost of the lowest cost proof as its confidence in answer option a_i. This is repeated for all answer options.

The Integer Linear Programming (ILP) solver

The ILP solver uses knowledge represented as a set of tables, akin to the classical relational model (Codd 1970) but built over natural language text. Each knowledge table T consists of a head-body pair (H_T, B_T) and captures a relation or predicate $R_T(x_1, \ldots, x_n)$ defined over entities. Columns of T represent attributes of R_T with attribute names in the

Country	*Location*
France	north hemisphere
USA	north hemisphere
...	
Brazil	south hemisphere
Zambia	south hemisphere
...	

Hemisphere	*Orbital Event*	*Month*
northern	summer solstice	Jun
northern	winter solstice	Dec
northern	autumn equinox	Sep
...		
southern	summer solstice	Dec
southern	autumn equinox	Mar
...		

Table 1: Examples of knowledge tables

header row H_T, rows of T represent n-tuples or instances of R_T, and cells of T represent entities as natural language phrases. Cohen (2000) studied noisy joins over similar tables, focusing on efficiently computing, given a database query, the top few matching data entries. Table 1 shows two (simplified) examples.

Tables were built using an interactive table-building tool applied to science texts. The tool performs bootstrapped relation extraction over a corpus, with a user in the loop to suggest syntactic patterns and accept/reject matches, allowing tables to be built quickly. Our eventual goal is to make table construction as automated as possible.

Informally, question-answering involves matching lexical chunks in the question q and answer option a_i against one or more table rows, or a chain of joined rows, and returning the strength of that match as the confidence in a_i. For example, a path through two joined rows in Table 1 strongly matches the question + answer "In USA, when is the summer solstice? (A) June". We call such connections, which in general form a connected graph structure, a *proof graph*, and question-answering involves selecting the answer option with the best scoring proof graph. Note that matching is not just string equality, as there may be lexical variation among question chunks and table cells (e.g., "fall" vs. "autumn").

Formally, we treat this task as a discrete global optimization problem over tables. We use the ILP formalism, which has been successful in several NLP tasks (Roth and Yih 2004; Srikumar and Roth 2011; Zhang, Hoffmann, and Weld 2012) but, to our knowledge, has not been applied directly to question-answering using semi-structured knowledge. Let $\mathcal{T}$ be a set of knowledge tables and $g(t, h)$ be a (possibly directional) similarity measure in $[0, 1]$ between short natural language phrases t and h. Let q denote a multiple choice question with answer options $A = \{a_i\}_i$ and correct answer $a^* \in A$. We build an ILP model $M = M(q, A, \mathcal{T})$ over a set V of integer-valued variables, a set C of linear constraints over V, and a linear maximization objective function $f = f(q, A, \mathcal{T})$ whose maximizer can be easily mapped to a candidate answer, ideally a^*. At a high level, M defines the space of possible proof graphs or reasoning patterns that "connect" lexical chunks of q with some answer option $a_i \in A$ through cells in one or more tables in $\mathcal{T}$.

Variables V define possible connections or edges between lexical chunks in q, answer options A, and cells of tables $\mathcal{T}$. To improve scalability and reduce noise, we use the top few (typically 4-6) tables matching q as ranked by a simple TF-IDF scoring mechanism.

Constraints C reduce the exponentially large search space to proof graphs to what a human might consider as meaningful and appropriate for elementary science reasoning. For instance, we enforce *structural constraints* such as: a proof graph P must have links to exactly one answer option, P may use at most k_1 rows per table and at least k_2 cells in any row it uses, etc. We also add *semantic constraints*, such as limiting pairs of columns in two tables that may be linked together, since not all table joins are meaningful. The constraints are designed from manual inspection of desirable proof graphs. Although manually built, they are not specific to 4th Grade Science so are a one-time cost.

The objective function f associates with every feasible proof graph P a score f_P designed to balance rewards (such as for including many links with high similarity measure g or linking with more columns of a table) with penalties (such as for using too many tables or low similarity links).

To score answer options, we first build the model $M(q, A, \mathcal{T})$ and solve it using an off-the-shelf ILP solver SCIP (Achterberg 2009). If a feasible solution is found, we read off the only answer option $a \in A$ that has links in the solution proof graph P, associate a with the value of f in P as its score, and repeat the above process with the remaining answer options $A \setminus \{a\}$ by disabling all links to a, until all answer options have been assigned a score or declared infeasible. The highest scoring option is then selected, while scores for the other options are used later in the Combiner.

Combination

Each solver outputs a non-negative confidence score for each of the answer options along with other optional features. The Combiner then produces a combined confidence score (between 0 and 1) using the following two-step approach.

In the first step, each solver is "calibrated" on the training set by learning a logistic regression classifier from each answer option to a correct/incorrect label. The features for an answer option i include the raw confidence score s_i as well as the score normalized across the answer options for a given question. We include two types of normalizations:

$$normal_i = \frac{s_i}{\sum_j s_j} \qquad softmax_i = \frac{\exp(s_i)}{\sum_j \exp(s_j)}$$

Each solver can also provide other features capturing aspects of the question or the reasoning path. The output of this first step classifier is then a calibrated confidence for each solver s and answer option i: $calib_i^s = 1/(1+\exp(-\beta^s \cdot f^s))$ where f^s is the solver specific feature vector and β^s the associated feature weights.

The second step uses these calibrated confidences as (the only) features to a second logistic regression classifier from answer option to correct/incorrect, resulting in a final confidence in $[0, 1]$, which is used to rank the answers:

$$confidence_i = 1/\left(1 + \exp\Big(-\beta^0 - \sum_{s \in Solvers} \beta^s \, calib_i^s\Big)\right)$$

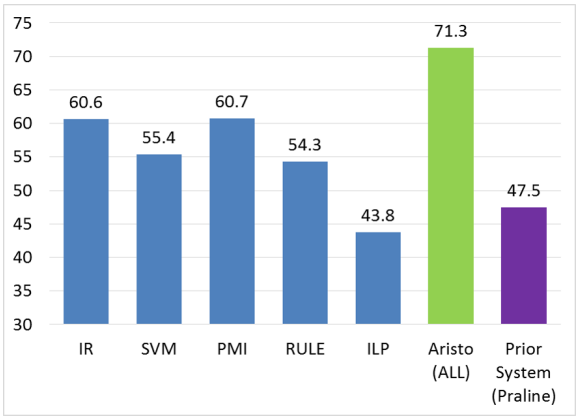

Figure 2: Aristo significantly outperforms all individual solvers and the best previously published system (Praline).

Here, feature weights β^s indicate the contribution of each solver to the final confidence. Empirically, this two-step approach yields more robust predictions given limited training data compared to a one-step approach where all solver features are fed directly into a single classification step.

Empirical Evaluation

We consider two questions: How well does STUDENT perform on 4th Grade Science problems; and do the different levels of representation all contribute to the overall score?

Dataset. We use real exam questions, exactly as written, from the NY Regents[1] 4th Grade Science exams.[2] We use all questions within our scope (no diagram, multiple choice, NDMC), using 6 years of exams (108 NDMC questions) for training and 6 years (129 NDMC questions) for testing, kept completely hidden during all stages of system development. Questions vary in length from roughly 8 to 70 words, and cover a wide variety of topics and styles. Although the low number of publically released, real exam questions makes the dataset small, it provides sufficient signal for evaluation.

Corpora, Rules, and Tables. We work with two corpora:

1. Elementary Science Corpus: 80k sentences about elementary science, consisting of a Regents study guide, CK12 textbooks,[3] and automatically collected Web sentences of similar style and content to that material.
2. Web Corpus: 5×10^{10} tokens (280 GB of plain text) extracted from Web pages.

From the Elementary Science corpus, a rulebase of 45,000 rules was automatically extracted for the RULE solver, and a datastore of 45 tables (10k rows, 40k cells) was built using an interactive table-building tool and manually, for the ILP solver. The PMI solver uses the Web Corpus and a window of 10 words to compute PMI values.

[1]Regents is the only State-level Board to make all prior exams publically available, making it an ideal choice.

[2]http://www.nysedregents.org/Grade4/Science/home.html

[3]www.ck12.org

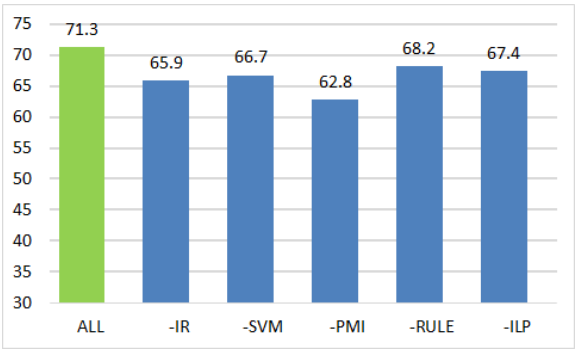

Figure 3: Effect of removing an individual solver from the ensemble. The results suggest each solver contributes to the overall score.

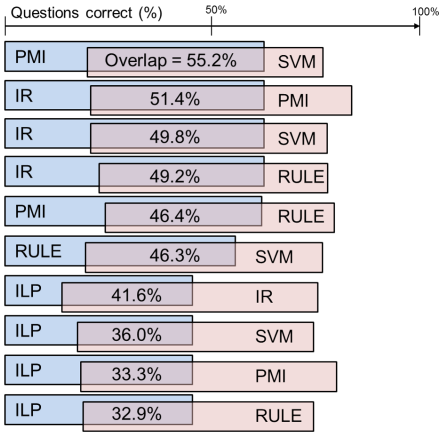

Figure 4: Different pairs of solvers answer substantially different question sets correctly (i.e., solvers are not redundant with each other). For example (line 1), of the questions PMI or SVM answer correctly, only 55.2% are in common.

Scoring A solver's score is the percent correct on the question set. If the solver produces N answers (N-way tie) including the correct one, it scores 1/N (equivalent to the asymptote of random guessing between the N). If no answer is produced, it is scored 1/K (for K-way multiple choice), equivalent to random guessing.

Results and Comparison. We used as baseline the Praline system of Khot et al. (2015), also developed for 4th Grade Science questions, and the best performing system in that prior work. Praline uses Markov Logic Networks (MLNs) (Richardson and Domingos 2006) to align lexical elements of questions and background science knowledge, and to control inference. Using its optimal parameter values, the highest score it obtained on our dataset was 47.5%.

We also ran our system, Aristo, as well as each individual solver, on this data. Aristo scored 71.3%, significantly (at the 95% confidence level) higher than both the previously published system Praline, as well as all the individual solvers (Figure 2). This suggests that the use of multiple levels of representation significantly improves performance on this class of problem.

To assess the contributions of individual solvers, we first

Representations	Solvers	Score (%)
Text	IR	60.6
Text+Statistical	IR+SVM+PMI	68.2
Text+Statistical+ Inferential	Aristo (ALL)	**71.3**

Table 2: Adding solvers using increasingly structured representations improves performance.

ablated each solver one by one. The results are shown in Figure 3. In all cases, the performance dropped with a solver removed, with the PMI solver contributing the most (-8.5% drop when ablated). This suggests that all solvers are contributing to the ensemble, and none are redundant. To further assess possible redundancy, we compared questions correctly answered by different solver pairs to see whether any two solvers were answering essentially the same questions, indicating possible redundancy in the ensemble. The degree to which solver pairs were answering the same questions correctly is shown in Figure 4, again indicating a high degree of non-redundancy.

To further assess the layers of representation, we compared Aristo with the text only layer (the IR solver), and the text and statistical layer (the IR, PMI, and SVM solvers) together. The results are shown in Table 2. As more layers of representation are added, performance improves, again suggesting value in using multiple representations.

Detailed Analysis

We provide insights into solvers' relative strengths and weaknesses based on a detailed analysis on the training data.

The PMI solver: This solver selects the answer that most (relatively) frequently co-occurs with the question words. There are two major causes of failure:

1. When there are multiple strong relationships between question words and answer options. For example, for:

   Which characteristic can a human offspring inherit? (A) facial scar (B) blue eyes (C) long hair (D) broken leg

   it selects (C) because "human" and "hair" have high co-occurrence, a distractor from "blue eyes" and "inherit".
2. When there are polarity-changing words (negation, good/bad, shortest/longest). For example, for:

   Which activity is an example of a good health habit? (A) watching television (B) smoking cigarettes (C) eating candy (D) exercising every day

   the solver selects (B) because "health" and "cigarettes" have high co-occurrence, overwhelming the co-occurrence between "good health" and "exercising".

Despite these limitations, our experiments suggest the PMI solver provides the strongest signal in Aristo.

The RULE solver: This solver performs well when the applied rule(s) express an important implication in the domain, and the entailment reasoning correctly connects the rule to the question. For example, for:

A turtle eating worms is an example of (A) breathing (B) reproducing (C) eliminating waste (D) taking in nutrients

the solver correctly infers answer (D) by applying the rule:

"IF animals eat THEN animals get nutrients"

extracted from the textbook, along with the knowledge that a turtle isa animal, and that "get" and "take in" are synonyms.

While this example illustrates how the solver can answer questions that other solvers struggle with, we observed two major failure modes:

1. Errors in the rulebase, question interpretation, or both, arising from poor syntactic analysis and interpretation.
2. Inadequacies in its heuristic scoring algorithm for computing the confidence that a rule applies.

The ILP solver: This solver's relative strength is for questions where multiple pieces of information need to be combined together. For example it successfully answers:

Which gas is given off by plants? (A) Hydrogen (B) Nitrogen (C) Oxygen (D) Helium

by joining two rows of two different tables together (expressed as propositions below):

table13:has-part("plant", "stomata").
tablez3:part-output("stomata", "oxygen").

In contrast, the PMI solver and the RULE solver both answer this question incorrectly: plants and nitrogen have high co-occurrence, and the RULE solver does not have an appropriate rule.

The curated table knowledge reduces the problem of noise (compared with the RULE solver), but there are still two main failure modes:

1. Missing knowledge: The curated tables do not cover the information required.
2. Control: While the table solver uses global constraints to constrain the paths explored, it may still find nonsensical paths (joins) through the tables during inference.

Difficult Questions

Despite Aristo's good performance, there are five classes of question that are hard for all of its constituent solvers to answer reliably, in the absence of a corpus sentence containing the answer explicitly:

1. Comparison questions
   ...moves faster or slower than ...?
2. Simple arithmetic reasoning
   7 rotations of Earth equals (A) 1 week (B) 2 weeks
3. Complex inference
   A white rabbit is best protected in (A) a snowy field..
4. Structured questions
   Which structure is correctly paired with its function?
5. Story Questions
   A puddle formed. Then the sun came out ...

In addition, linguistic variation, e.g., *"get a better look at/view in more detail", "removal of heat"/"drop in temperature", "35F"/"cold"*, is a challenge in all questions.

Conclusion

Fourth Grade Science tests are difficult for machines due to the wide variety of knowledge and reasoning requirements, making it a unique challenge for the community. In this paper we have presented Aristo, a system that addresses this challenge by using a family of representations with different levels of structure. The significance of this work is three-fold. First, the system achieves a new level of performance on this task compared with previously published work, demonstrating value in the approach. Second, our failure analysis illuminates the complementary strengths of the different methods, and where future work is needed. Finally, although Fourth Grade test-taking is not itself an application, we have shown that it requires QA that goes significantly beyond retrieval techniques, yet is simple enough to be accessible. In this sense, it is a simple embodiment of an AI challenge requiring both language and reasoning, with many potential applications if solved (e.g., tutoring systems, a scientist's assistant), and hence worthy of further exploration. We are releasing our datasets (at www.allenai.org) to encourage such research.

Acknowledgements

We are indebted to Paul Allen whose long-term vision inspired this project, and continues to inspire our scientific endeavors. We are grateful to Peter Jansen and Mihai Surdeanu who created the SVM solver and provided the implementation. We are also grateful to Niranjan Balasubramanian, Sumithra Bhakthavatsalam, Isaac Cowhey, Dirk Groeneveld, Kevin Humphreys, Jesse Kinkead, Roie Levin, Carissa Schoenick and Sam Skjonsberg for their contributions to this work.

References

Achterberg, T. 2009. SCIP: solving constraint integer programs. *Mathematical Programming Computation* 1(1):1–41.

Brill, E.; Dumais, S.; and Banko, M. 2002. An analysis of the AskMSR question-answering system. In *Proceedings of EMNLP*, 257–264.

Church, K. W., and Hanks, P. 1989. Word association norms, mutual information and lexicography. In *27th ACL*, 76–83.

Clark, P., and Etzioni, O. 2016. My Computer is an Honor Student but how Intelligent is it? Standardized Tests as a Measure of AI. *AI Magazine*. (To appear).

Clark, P.; Balasubramanian, N.; Bhakthavatsalam, S.; Humphreys, K.; Kinkead, J.; Sabharwal, A.; and Tafjord, O. 2014. Automatic construction of inference-supporting knowledge bases. In *4th Workshop on Automated Knowledge Base Construction (AKBC)*.

Clark, P. 2015. Elementary school science and math tests as a driver for AI: take the Aristo challenge! In *29th AAAI/IAAI*, 4019–4021.

Codd, E. F. 1970. A relational model of data for large shared data banks. *Communications of the ACM* 13(6):377–387.

Cohen, W. W. 2000. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems* 18(3):288–321.

Fader, A.; Zettlemoyer, L.; and Etzioni, O. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of SIGKDD*, 1156–1165.

Ferrucci, D.; Brown, E.; Chu-Carroll, J.; Fan, J.; Gondek, D.; Kalyanpur, A. A.; Lally, A.; Murdock, J. W.; Nyberg, E.; Prager, J.; et al. 2010. Building Watson: An overview of the DeepQA project. *AI magazine* 31(3):59–79.

Gunning, D.; Chaudhri, V.; Clark, P.; Barker, K.; Chaw, J.; and Greaves, M. 2010. Project Halo update - progress toward digital Aristotle. *AI Magazine* 31(3).

Hosseini, M. J.; Hajishirzi, H.; Etzioni, O.; and Kushman, N. 2014. Learning to solve arithmetic word problems with verb categorization. In *2014 EMNLP*, 523–533.

Jansen, P.; Surdeanu, M.; and Clark, P. 2014. Discourse complements lexical semantics for non-factoid answer reranking. In *52nd ACL*, 977–986.

Khot, T.; Balasubramanian, N.; Gribkoff, E.; Sabharwal, A.; Clark, P.; and Etzioni, O. 2015. Exploring Markov logic networks for question answering. In *2015 EMNLP*.

Ko, J.; Nyberg, E.; and Si, L. 2007. A probabilistic graphical model for joint answer ranking in question answering. In *Proceedings of SIGIR*, 343–350.

Kushman, N.; Zettlemoyer, L.; Barzilay, R.; and Artzi, Y. 2014. Learning to automatically solve algebra word problems. In *52nd ACL*, 271–281.

Mikolov, T.; Karafiát, M.; Burget, L.; Cernockỳ, J.; and Khudanpur, S. 2010. Recurrent neural network based language model. In *INTERSPEECH*, 1045–1048.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Novak, G. 1977. Representations of knowledge in a program for solving physics problems. In *IJCAI-77*.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62(1–2):107–136.

Roth, D., and Yih, W. 2004. A linear programming formulation for global inference in natural language tasks. In *CoNLL*, 1–8. ACL.

Salton, G. 1971. *The SMART retrieval system-Experiments in automatic document processing*. Prentice-Hall, Inc.

Seo, M. J.; Hajishirzi, H.; Farhadi, A.; and Etzioni, O. 2014. Diagram understanding in geometry questions. In *28th AAAI*, 2831–2838.

Srikumar, V., and Roth, D. 2011. A joint model for extended semantic role labeling. In *EMNLP*.

Surdeanu, M.; Ciaramita, M.; and Zaragoza, H. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics* 37(2):351–383.

Töscher, A.; Jahrer, M.; and Bell, R. M. 2009. The BigChaos solution to the Netflix Grand Prize. *Netflix prize documentation*.

Yao, X., and Van Durme, B. 2014. Information extraction over structured data: Question answering with Freebase. In *52nd ACL*.

Yih, W.-t.; Chang, M.-W.; Meek, C.; and Pastusiak, A. 2013. Question answering using enhanced lexical semantic models. In *Proc. ACL*.

Zhang, C.; Hoffmann, R.; and Weld, D. S. 2012. Ontological smoothing for relation extraction with minimal supervision. In *26th AAAI*.

Zou, L.; Huang, R.; Wang, H.; Yu, J. X.; He, W.; and Zhao, D. 2014. Natural language question answering over rdf: a graph data driven approach. In *Proceedings of SIGMOD*, 313–324.